FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Serviço para Product Placement em Televisão

André Casais Regado

Mestrado Integrado em Engenharia Informática e Computação

Orientador: Prof. Ricardo Santos Morla

Coorientador: Prof. Ademar Manuel Teixeira de Aguiar

Supervisor: Alexandre Ulisses F. Almeida e Silva

24 de Julho de 2017

Serviço para Product Placement em Televisão

André Casais Regado

Mestrado Integrado em Engenharia Informática e Computação

Aprovado em provas públicas pelo Júri:

Presidente: Professor Gabriel de Sousa Torcato David
Vogal Externo: Professor Sérgio Armindo Lopes Crisóstomo
Orientador: Professor Ricardo Santos Morla

---

24 de Julho de 2017

Resumo

Ao longo dos últimos anos o aumento do consumo de vídeos sobre formato digital tem sido notório, ou seja, cada vez mais as pessoas assistem televisão a partir dos seus *tablets, smartphones* e portáteis. O que possibilita, devido ao acesso à Internet, uma personalização da publicidade apresentada ao consumidor final.

A forma de apresentar publicidade em vídeo está a mudar. Se antes a utilização dos chamados *Ad Breaks* eram uma evidência, hoje em dia, já não se pode afirmar o mesmo, pois têm-se refletido ineficazes e ineficientes. Surge assim uma nova e mais eficiente maneira de fazer publicidade em vídeo, *Native In Video Advertising* (NIVA), onde a publicidade fará parte do programa o que levará que o visualizar não consiga rejeitar as mensagens publicitárias.

Atualmente, não há ninguém que ofereça um vídeo transmitido com diferentes produtos dentro do programa (NIVA) baseados na localização do visualizador, o que nos permite desenvolver um projeto inovador. Para isso, serão guardados pequenos segmentos do programa com diferentes produtos neles inseridos, isto porque, NIVA é computacionalmente bastante exigente. A contextualização da publicidade baseada na localização poderá ser um método bastante eficiente, pois irá permitir às marcas que se dirijam a possíveis clientes na altura certa.

Esta dissertação aborda um novo serviço de *product placement* em televisão, propondo uma arquitetura em que o objetivo é a criação de uma plataforma que possibilite a inserção de segmentos personalizados, ou seja, incorporados com anúncios, utilizando uma técnica de *Adaptive Bit rate Streaming*, mais propriamente, o MPEG-DASH.

O protótipo desenvolvido permite construir o vídeo final com os anúncios mais indicados baseados na localização do utilizador e, a partir, de uma interface WEB será possível visualizar esse mesmo vídeo.

Abstract

The increase of video consumption under digital format is evident. Nowadays everyone has a smartphone or tablet, where they can watch television. Usually, these devices have Internet access which allows us to contextualize ads according the end user.

The way to display ads in video are changing. Conventional advertising Ad Breaks are dying because they became inefficient. Native Advertising is the new way to display advertising, where the ads will be embedded in the video content so that the viewer can't reject the advertisements' messages.

Nowadays, no one offer different objects placed in video (NIVA) based on user's network location which allows us develop a new and innovate project.To achieve this, we will have a repertoire of small segments of the programme with different products placed in them, this is because producing NIVA advertising is computationally intensive. Personalized adverts based on location can be efficient because they will permit to reach possible clients at the precise moment.

This thesis presents a high-level architecture for product placement in television, where the goal is to create a platform that allows an insertion of personalized segments, in real time, using Adaptive Bitrate Streaming, namely, MPEG-DASH.

The viewer will see the program in real time, but the architecture will be switching streams seamlessly between main program scenes, selected stream of program with most appropriate item placed in it and back to the main program stream. From a WEB interface it will be possible view the final video.

Agradecimentos

Em primeiro lugar, gostaria de agradecer aos meus pais por me terem proporcionado as melhores condições para a minha formação tanto profissional como pessoal. Também quero agradecer a todos os meus amigos que me ajudaram e fizeram parte da minha vida nestes últimos anos.

À MOG Technologies por me terem dado todas as condições para o desenvolvimento do trabalho e um especial agradecimento ao Eng. Pedro Santos e ao Eng. Miguel Poeira, que me acompanharam mais de perto neste trabalho e ao Eng. Alexandre Ulisses pela disponibilidade demonstrada e por todo o conhecimento transmitido.

A todos os envolvidos na minha formação na FEUP, em especial, ao meu orientador e coorientador, Professor Ricardo Morla e o Professor Ademar Aguiar.

André Regado

"I have not failed. I've just found 10,000 ways that won't work."

Thomas Edison

Conteúdo

Lista de Figuras

Lista de Tabelas

Abreviaturas e Símbolos

API	Application Programming Interface
CDN	Content Delivery Network
COD	Common Object Descriptor
CPU	Central Processing Unit
CSS	Cascading Style Sheets
DASH	Dynamic Adaptive HTTP Streaming
DTD	Document Type Definition
EDL	Edit Decision List
GPS	Global Positioning Systems
GUI	Graphical user interface
HD	High Definition
HDS	Adobe HTTP Dynamic Streaming
HLS	Apple HTTP Live Streaming
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
IP	Internet Protocol
LBA	Location Based Advertising
LBS	Location Based Services
MMS	Microsoft Smooth Streaming
MPD	Media Presentation Description
MSXML	Microsoft XML Core Services
MXF	Material Exchange Format
REST	Representational State Transfer
RPlayer	Retocatable Player
RTMP	Real-Time Messaging Protocol
RTMPT	Real-Time Messaging Protocol
TCP	Transmission Control Protocol
TS	Transport Stream
UDP	User Datagram Protocol
URL	Uniform Resource Locator
VAST	Video Ad Serving Template
VMAP	Video Multiple Ad Playlist
W3C	World Wide Web Consortium
XML	Extensible Markup Language
XSD	XML Schema

Capítulo 1

Introdução

Neste primeiro capítulo será apresentada uma visão geral da dissertação, incluindo os objetivos deste trabalho bem como as suas motivações. Uma breve análise do projeto a desenvolver também será apresentada, bem como a estrutura do restante documento da dissertação.

1.1 Contexto

A publicidade é uma grande fonte de rendimento de muitos serviços. A publicidade em vídeo está em mudança onde o objetivo é ir ao encontro das exigências e da satisfação do consumidor, do anunciante (pretende atingir o público certo e prender a atenção do visualizador na publicidade) e pelo responsável pela transmissão do vídeo (pretende obter rendimentos da publicidade, mas, ao mesmo tempo, não tenciona perder visualizadores).

A ineficácia demonstrada pelos anúncios tradicionais, os *Ad Breaks*, levou ao surgimento de um novo termo *Native In Video Advertising* (NIVA), este tipo de publicidade na televisão fará com que a publicidade faça parte do conteúdo do programa. Juntamente a isso, poderá ser feita uma contextualização da publicidade consoante o consumidor final. Já existem inúmeras formas de o fazer, como por exemplo, baseado no perfil ou pesquisas do utilizador. Contudo não existe ninguém a oferecer um vídeo transmitido com diferentes produtos dentro do programa (NIVA) baseados na localização do utilizador, o que irá permitir desenvolver um produto novo e inovador.

A forma de transmissão de vídeo, ao longo dos últimos anos, têm evoluído bastante, pois cada vez mais, os consumidores são mais exigentes, onde a técnica mais utilizada e mais poderosa hoje em dia é *adaptive bitrate streaming*.

De modo a se perceber melhor o problema, irá ser apresentado o modelo de negócio desta solução. Na figura 1.1 é possível ver um exemplo de um possível negócio. O *McDonald's* como muitos outros locais, oferecem *Wi-fi* gratuito aos seus clientes. O objetivo passa por estes locais, como o *McDonald's* pagarem à empresa pelo serviço, para que quando os seus clientes estiverem

a visualizar televisão a partir do *Wi-fi* disponibilizado por eles, serem transmitidos anúncios personalizados. Esses anúncios personalizados poderão ser ou anúncios da própria marca ou então anúncios de uma outra marca que lhes irá pagar para mostrarem anúncios dessa mesma marca.

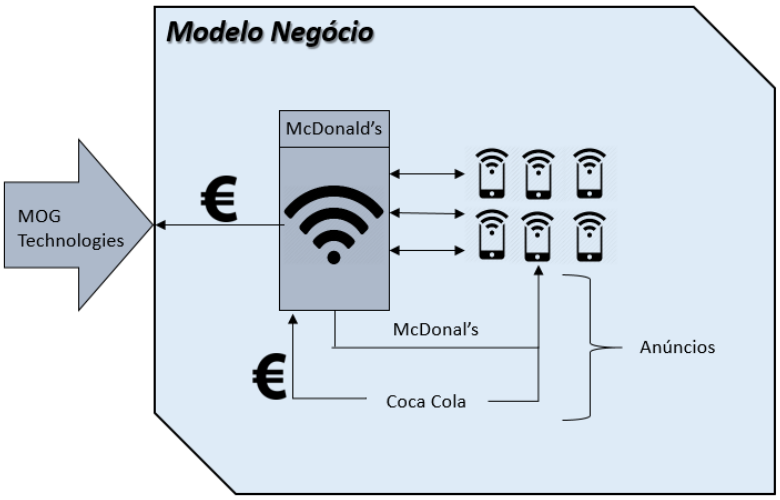

Figura 1.1: Exemplo do Modelo de Negócios

1.2 Enquadramento

Esta dissertação foi realizada em ambiente empresarial, na empresa MOG Technologies, especializada no mercado de *broadcasting* há mais de 10 anos, ou seja, foca-se na distribuição de conteúdos, como vídeo e áudio.

Esta dissertação enquadra-se na área de desenvolvimento de software da empresa, com o intuito de desenvolver um novo produto e, sendo importante, a utilização de ferramentas já existentes da *MOG Technologies* o que irá permitir, facilmente, a inserção desta solução com os atuais produtos da empresa. Existe o potencial e capacidade para esta solução correr na *cloud*, será única no mercado e será totalmente compatível com os restantes produtos da empresa.

1.3 Motivação e Objetivos

O aumento da visualização de vídeos sob formato digital é evidente. Hoje em dia, qualquer pessoa tem um *smartphone* ou um *tablet* onde pode ver televisão, ou seja, dispositivos estes que terão acesso à Internet. O que permite contextualizar os anúncios mostrados ao utilizador final, neste caso, consoante a sua localização.

A forma de fazer publicidade está em mudança. A forma mais comum de visualizar publicidade em vídeo são os chamados *Ad Breaks* (pre-rolls, mid-rolls, post-rolls), ou seja, há intervalos de tempo no programa para mostrar publicidade contudo esta forma de fazer publicidade torna-se incómoda para o espetador e, consequentemente, ineficaz e ineficiente. Daí surgir uma nova

forma de fazer publicidade em vídeo, chamada *Native In Video Advertising*, em que o principal objetivo é fazer com que a publicidade faça parte do programa e, ao mesmo tempo, torne o vídeo mais realista. Daí surgir a oportunidade de desenvolver uma plataforma que possibilite a inserção de segmentos personalizados consoante a localização do utilizador final. Será usado técnicas de *Adaptive Bitrate Streaming*, em particular o MPEG-DASH para transmitir o vídeo final ao utilizador.

O trabalho a realizar visa a desenvolver um protótipo de uma aplicação que permita a troca entre o vídeo original e os anúncios contextualizados a inserir no vídeo. O vídeo a mostrar ao utilizador é baseado na sua localização. Sendo que, esse vídeo será fornecido ao utilizador em formato DASH. Adicionalmente, deve ser possível visualizar este protótipo através de uma aplicação web. Também é importante referir que foi escrito um artigo científico sobre este trabalho, podendo encontra-lo nos anexos A.1, onde permitiu-me participar numa conferência *TVX 2017 (ACM International Conference On Interative Experiences For Television and Online Video)* em Amesterdão, nomeadamente no *Workshop: In-Programme Personalisation for Broadcast* [1] .

De modo a desenvolver a aplicação proposta definiram-se os seguintes objetivos:

- Estudo da meta informação que a *MOG Technologies* recebia, que continham os dados sobre os anúncios, nomeadamente, o VMAP e VAST.
- O software desenvolvido deverá ser integrável com as arquiteturas e produtos atuais da empresa.
- Desenvolvimento de um módulo *Business Logic* que receba informação sobre os anúncios, criando assim, uma estrutura de informação num formato aceite por uma biblioteca da MOG.
- Desenvolvimento de uma aplicação web que permita distinguir a localização do cliente, a partir, do seu endereço IP e, deste modo, conseguir saber qual vídeo mostrar ao cliente. Esta aplicação também comunica com uma ferramenta da MOG, o RPlayer, de modo criar DASH do vídeo a mostrar ao cliente.
- Implementação de uma Interface Web que construirá um cenário de visualização de um vídeo em formato DASH.
- Escrita de um artigo científico, possibilitando assim, a participação numa conferência em Amesterdão.

1.4 Estrutura da Dissertação

Para além da introdução, esta dissertação contém mais 6 capítulos.

[1] https://tvx.acm.org/2017/

- No capítulo 2, é abordado o tema publicidade em televisão. Sendo importante entender o porquê da mudança na forma de fazer publicidade em vídeo, bem como as suas vantagens. Também foram evidenciados os tipos de *product placement* existentes. Além disso, e ainda neste capítulo, foi referido o tema da publicidade personalizada, nomeadamente, a publicidade baseada na localização.

- O capitulo 3, é explicado a evolução na forma de *streaming* ao longo dos últimos anos até aos dias de hoje. Sendo que, hoje em dia, a técnica de *adaptive bitrate streaming* é a mais frequente, foi analisado os quatro tipos de implementações existentes.

- Neste capitulo 4 são apresentados e descritos trabalhos relacionados, apontando-se as suas limitações e diferenças.

- O capítulo 5 propõe e descreve uma arquitetura para a aplicação proposta. Com base na arquitetura proposta é descrito *workflow*, bem como uma explicação sobre a implementação. São enumeradas todas as ferramentas auxiliares que contribuíram para o desenvolvimento desta aplicação.

- O capitulo 6 apresentada um conjunto de testes de validação de forma a validar o protótipo desenvolvido, analisando-se os resultados obtidos desses testes.

- A dissertação termina com o capítulo 7, onde é feita uma apreciação geral do trabalhado realizado, o cumprimentos dos objetivos e das possibilidades de trabalho futuro.

- Por fim, nos anexos, é possível encontrar o artigo científico e exemplos de ficheiros utilizados na implementação e testes.

Capítulo 2

Publicidade

O desenvolvimento tecnológico trouxe-nos inúmeras vantagens entre as quais a possibilidade de as marcas mostrarem os seus produtos aos consumidores na televisão de uma forma mais eficiente e eficaz. Sendo isto, importantíssimo, visto que a publicidade é fundamental no modelo de negócios de muitos serviços sendo, por vezes, a maior fonte de rendimento.

A forma comum de aparecer publicidade na televisão eram os chamados *ad breaks* em *pre-rolls*, *mid-rolls* ou *post-roll*, isto significa, que havia uma paragem entre os segmentos do programa da televisão para uma marca expor o seu produto e dar a conhecer às pessoas, podendo ser no inicio, no meio ou no fim do vídeo. Contudo, esse método foi visto como ineficaz e ineficiente por algumas razões, como: durante esse intervalo as pessoas aproveitavam para fazer outras coisas, como por exemplo, ir à casa de banho ou apenas desviam a sua atenção quando este tipo de publicidade aparecia.

Os formatos de anúncios no vídeo mais comuns são [IAB16] :

- *Linear Ads*: este tipo de anúncios é usado antes, no meio ou no fim do vídeo e normalmente são usadas imagens ou vídeos, que interrompem o vídeo.

- *Nonlinear Ads*: este tipo de anúncios consistem em sobrepor uma imagem sobre o vídeo que está a a correr. Normalmente aparecem no topo ou em baixo do vídeo e duram cerca de dez a vinte segundos.

- *Companion Ads*: é um *banner* que permanece no vídeo depois de um *Ad Break* com o objetivo de reforçar a publicidade à marca.

- *Ad Podes*: é um conjunto de anúncios lineares sequenciais.

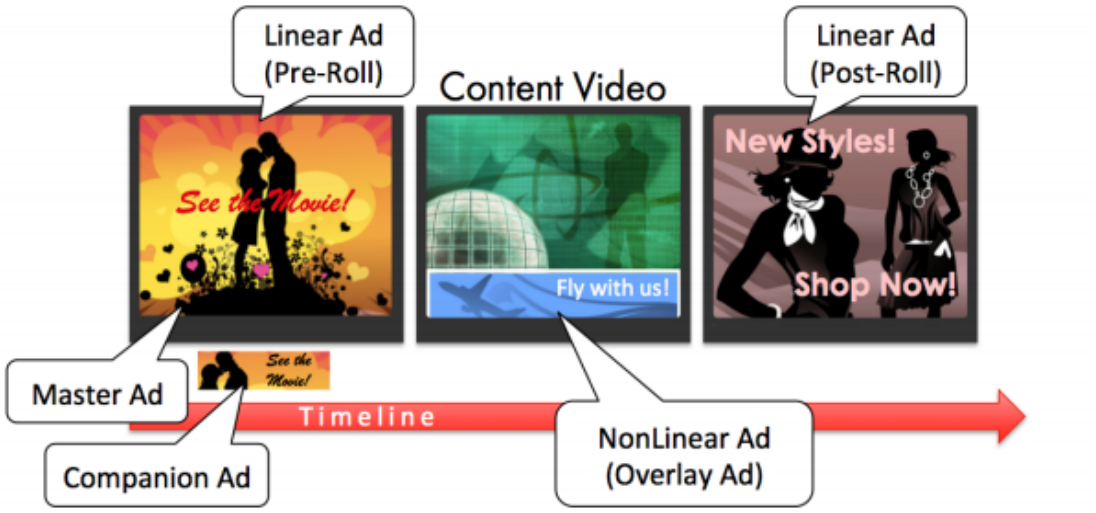

Figura 2.1: Tipos de anúncios em vídeo [IAB16]

Um dos problemas que se têm enfrentado ultimamente na publicidade em vídeo, é o crescimento do uso dos *ad blockers*, ou seja, hoje em dia, é, facilmente, possível bloquear os *Ad Breaks* no vídeo. Um *ad blocker*, como o nome indica, é um *software* que previne os anúncios serem entregues ao visualizador. Os *ad blockers*, tipicamente, são implementados como *plugins* que são instalados e intercetam e eliminam pedidos de anúncios. Usam uma variedade de mecanismos para identificar esses pedidos, em que o mais comum e o mais eficiente baseia-se na comparação dos URLs que estão incorporados nos pedidos com uma lista de URLs de servidores de anúncios [MMCB16, WZX+16].

Devido a necessidade de atingir o público, sem o aborrecer e irritar, surge assim o termo *native advertising*, cujo a ideia é as marcas fazer propaganda aos seus produtos de uma forma indireta. Ou seja, a publicidade estaria presente, mas o espetador não iria interpretar aquilo como sendo publicidade e de uma forma que não o perturbasse tanto. Resumidamente, em vez de a publicidade aparecer de uma forma separada que faz com que as pessoas se incomodem, a publicidade fará parte do conteúdo. Esta adição de publicidade em segmentos do programada de televisão deverão parecer realistas pois sem o parecer tirará prazer aos utilizadores em visualizar aquele programa e, consequentemente, os espetadores não irão ver. Normalmente, existe uma preferência por parte dos espetadores ver um programa de televisão onde aparecem objetos neles pois provoca no espetador uma sensação de ser mais natural [Pul14].

2.1 *Product Placement*

A figura 2.2 mostra-nos que a tendência do uso da *Native Advertising* está a aumentar e irá aumentar nos próximos anos, o que levará a uma maior receitas a partir desse tipo de publicidade ao contrário da publicidade não nativa [Bus].

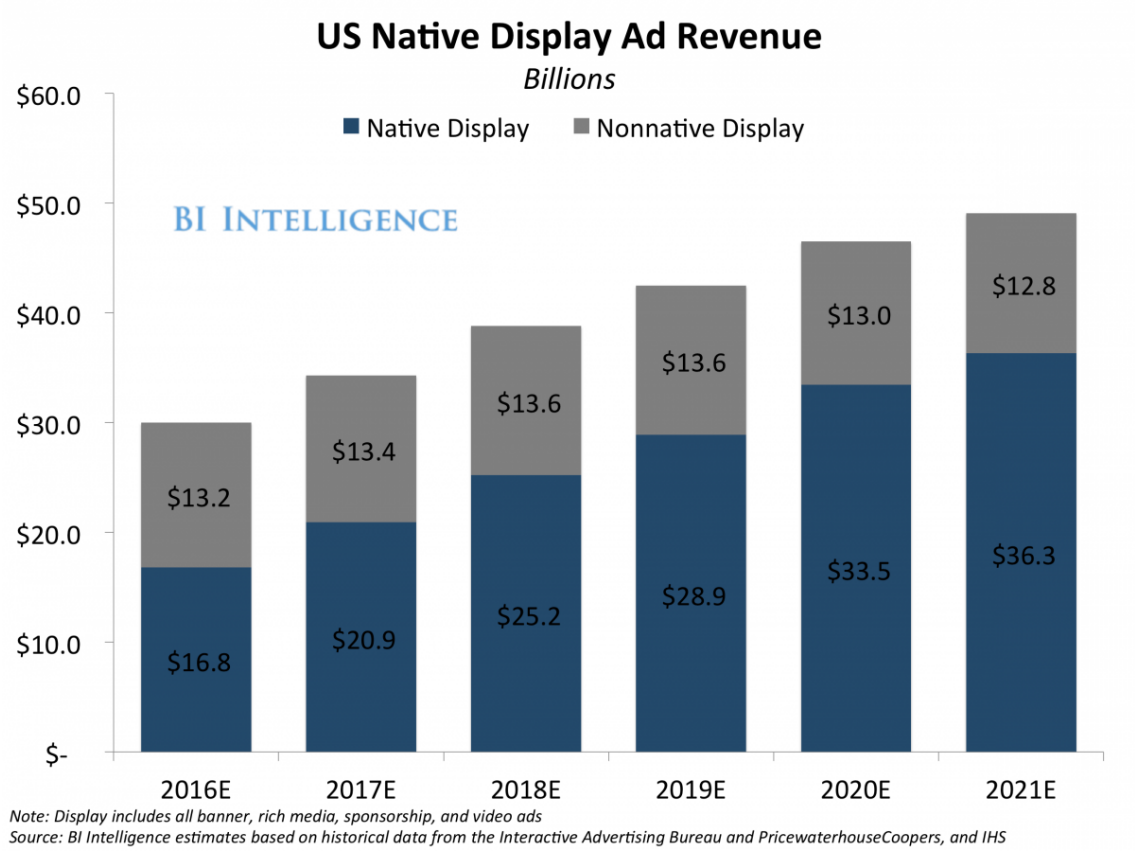

Figura 2.2: Previsão de receitas de Native vs Nonnative Advertising [Bus]

Product placement é uma técnica de publicidade usada por empresas para promover subtilmente os seus produtos ou marca, ou seja, não é envolvido anúncios diretamente mas torna marcas, produtos e serviços visíveis. Esta técnica é muito usada na televisão e nos filmes.

Quando se está a visualizar um vídeo, o normal é que toda a atenção do visualizador esteja sobre aquilo que se desenrola à sua frente. E, automaticamente, o espetador ao ver um vídeo experiência algo e cria uma relação com aquilo que ouve e vê. Essa relação é concretizada, podendo tanto ser positiva ou negativa, dando lugar à possibilidade de criar laços com uma eventual marca ou produto que surja associado. A colocação estratégica dos produtos é feita com o intuito de criar uma boa ligação sobre eles. Para além do facto, de quem produz o vídeo, o *product placement* estabelece credibilidade quando apresenta objetos ou marcas que correspondem à realidade, contribuindo assim de forma positiva para a progressão da história. No entanto, quer se garantir a visibilidade do produto, porém se não houver cuidado com a integração do mesmo, o aparecimento deste no ecrã pode estranhar-se e revelar-se assim ineficaz e frustrante.

Product Placement pode aparecer de três diferentes formas [KD12, KD]:

1. *Passive placement*: ocorre quando um produto, serviço ou logo podem ser observadas.

2. Verbal: ocorre quando um produto, serviço ou uma empresa são mencionadas, ou seja, há uma referência verbal.

3. *Active placement*: ocorre quando um ator/atriz interage ou lida com um produto, serviço ou empresa.

Existe quatro tipos de *product placement* [KD12, KD]:

- *Classic placement*: é a mais simples, pois a preocupação é apenas fazer o produto ou marca aparecer no ecrã. A grande vantagem deste tipo de *product placement* é o facto de ser simples e o que têm custos mais reduzidos. Contudo, se houver um elevado número de inserções no mesmo vídeo, este passa facilmente despercebido.

- *Corporate placement*: existe uma prioridade na marca sobre o produto. A grande desvantagem deste tipo de *product placement* é o facto de passar despercebido, só se assumir que os visualizadores já conhecem o nome da marca antes de ver o vídeo, contudo é facilmente exposto no ecrã, principalmente no pós-produção.

- *Evocative placement*: é uma operação mais discreta, na medida em que a marca não aparece no ecrã. A grande desvantagem deste tipo de *product placement* é, também, o facto de não ser facilmente reconhecida pelos espetadores que não conhecem a marca. A grande vantagem é esta ser mais subtil do que a clássica.

- *Stealth placement*: extremamente discreta, sendo que está totalmente integrada na cena e não cria qualquer tipo de obstrução. É facilmente passada por despercebida mas a grande vantagem é que esta publicidade é perfeitamente integrada na história da cena.

Na figura seguinte 2.3, é possível observar um exemplo de cada tipo de *product placement*. Na primeira imagem, encontra-se um exemplo do *classic placement* onde pode ser observado um computador Vaio da Sony utilizado pela personagem de James Bond em Casino *Royale*. Na seguinte podemos observar um exemplo de *corporate placement*, em *Minority Report*, surge um cartaz publicitário da *GAP*, não evocando produtos concretos. Na terceira, é um exemplo *evocative placement*, onde no filme *Forrest Gump*, onde numa referência a uma suposta "companhia de frutas"é mostrado o logo da *Apple* na carta que Tom Hanks segura. E por último, observamos um exemplo de *stealth placement* onde *Kirsten Scott Thomas*, no filme *The Horse Whisperer*, é vestida pela *Calvin Klein*.

Figura 2.3: Tipos de *product placement*

Existem várias vantagens inerentes ao *product placement*. Uma das razões para os produtos estarem no vídeo é uma forma de reduzir os custos de produção. Outra razão é o facto de estes darem um aspeto de realismo porque um vídeo sem marcas pode parecer irrealista. No entanto, o objetivo principal com o *product placement* por parte da vista das empresas é aumentar a consciência da marca. Comparando com a publicidade tradicional na televisão, a principal vantagem é o facto de o espetador não conseguir evitar a exposição ao produto ou marca quando estão a ver televisão [AL12].

Contudo a *Product Placement* traz um problema complexo, porque cada segmento do programa de televisão têm uma forma única de apresentar o seu conteúdo. É necessário o uso de técnicas de visão por computador para a incorporação dos objetos no vídeo de uma forma subtil, ou seja, de uma forma em que o telespetador ao visualizar um vídeo, lhe pareça tudo natural.

2.2 Anúncios Personalizados em vídeo

Existem dois tipos de abordagens: anúncios não personalizados e anúncios personalizados. Anúncios não personalizados foi uma abordagem utilizada nos sistemas iniciais que consistia em mostrar os mesmos anúncios a todos os utilizadores, enquanto os anúncios podem ser personalizados consoante o utilizador que os vai receber, que irá permitir satisfazer o utilizador, o anunciante e quem permite a introdução dos anúncios no seu vídeo. Esta personalização pode ser dividida em três categorias [SM12]:

- *Sponsored Search Advertising*: esta forma de personalização de anúncios é baseada em pesquisas do utilizador, sendo utilizada pelos principais motores de busca como a Google, Yahoo and Microsoft.
- *Contextual Match Advertising*: esta forma de personalização de anúncios relaciona o conteúdo do anúncio com o conteúdo do que o utilizador esta a ver.
- *Shopping Website*: esta forma de personalização de anúncios é baseada em compras já feitas pelo utilizador ou então a partir da informação demográfica do utilizador. [CLWH16].

2.3 Publicidade baseada na localização (LBA)

A publicidade baseada na localização surge pelo facto de, hoje em dia, a qualquer sítio que vamos, levamos um dispositivo móvel connosco. Usando os dados da localização de uma pessoa, recolhida pelos dispositivos móveis, os anunciantes podem enviar diferentes mensagens às pessoas dependendo de onde elas se encontrem [DC15].

O uso da localização para a publicidade é eficiente porque: primeiro é personalizado, podendo a publicidade ser personalizada de acordo com o local onde as pessoas se encontram. Segundo é oportuno, isto porque os dados da localização são em tempo real, surgindo assim a oportunidade de as marcas se dirigirem às pessoas num preciso momento.

2.3.1 Serviços baseados na localização (LBS)

LBS (*Location Based Services*) é um serviço que permite saber onde o dispositivo do utilizador está localizado [KK11].

2.3.1.1 Componentes LBS

Para um utilizador poder utilizar um serviço baseado em localização, serão necessários quatro componentes [AU14] :

- Dispositivos Móveis: é o que o utilizador usa para fazer o pedido da informação. Podendo ser, por exemplo: telemóveis ou portáteis.

- Rede de comunicação: responsável por transferir dados do utilizador e do pedido ao prestador de serviços e, posteriormente, transferir a informação do pedido para o utilizador.

- Posicionamento: responsável por localizar a posição do utilizador. Poderá ser usado, por exemplo, o GPS (*Global Positioning systems*).

- Fornecedor do serviço e da aplicação: responsável por processar o pedido do utilizador. Este serviço calcula a posição, procura informação relacionada com o posicionamento ou até procura informação acerca de objetos de interesse do utilizador.

- Fornecedor de dados e conteúdo: toda a informação requisitada pelos utilizadores não são armazenados e mantidos no prestador de serviços. Sendo que, informação geográfica e de um local terá de ser pedida a um parceiro.

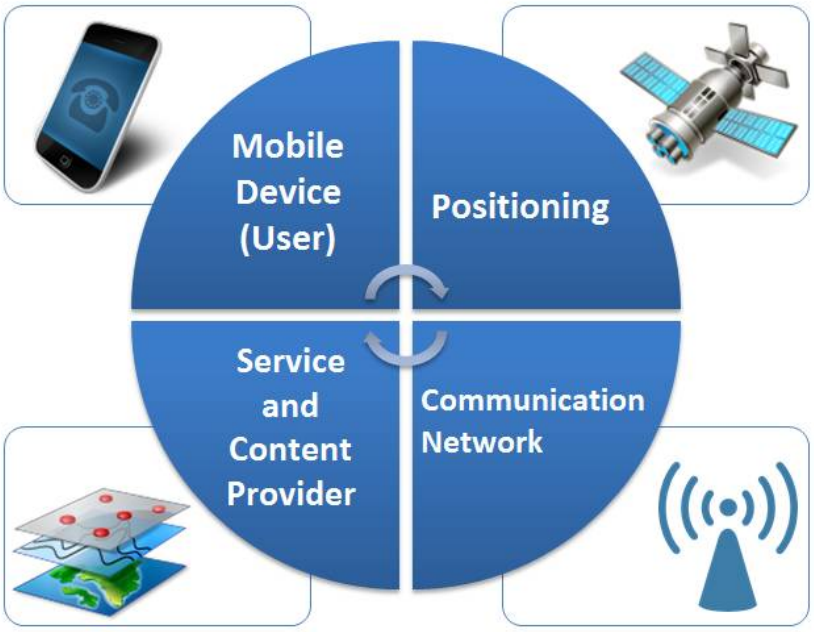

Figura 2.4: Componentes do LBS [geo]

2.3.1.2 Características de LBS

LBS têm um conjunto de características [RDWG08]:

- Há dois tipos de aplicações sobre a entrega de informação:
  1. Serviços do tipo *Push*: este tipo de serviço fornece informações ao utilizador sem este o ter requisitado. Isto acontece, por exemplo, com a publicidade, quando um desconto é enviado par ao utilizador de um restaurante na área em que ele se encontra.
  2. Serviços do tipo *Pull*: este tipo de serviço é quando o utilizador usa uma aplicação e pede informações à rede. Um exemplo deste tipo de serviços é, por exemplo, procurar pelo cinema mais próximo.
- Há duas maneiras de se obter informações sobre o utilizador
  1. Modo Direto: o perfil do utilizador é obtido diretamente pelo o mesmo.
  2. Modo Indireto: o perfil do utilizador é obtido por terceiros ou é analisado um padrão de interações do utilizador.
- Há três maneiras de interação entre o utilizador e o fornecedor do serviço:
  1. Ambos, o utilizador e o fornecedor de serviço, utilizam o *smartphone's*, como é exemplo aplicações como encontrar um amigo.

2. Só o utilizador é que utiliza *smartphone's*, como são exemplos de aplicações de contextualização de publicidade e rastreio do veículo.
3. Só o fornecedor de serviços é que utiliza *smartphone's* como o *check-in* automático do aeroporto.

2.3.2 Tipos de Aplicações que utilização a localização

Existe uma grande quantidade de serviços baseados na localização que podem ser agrupados nas seguintes categorias [Buc12, SWY08]:

- *Marketing*: utilização da localização do utilizador, por parte das empresas, para mandar mensagens de texto ou até de multimédia. Existem vários exemplos, mas um mais evidente é o envio de mensagens de promoções, e com validade, por parte das empresas a potenciais clientes que se encontrem perto dos seus estabelecimentos, podendo assim, aumentar a eficiência da publicidade. A característica acentuado é a utilização do serviço do tipo *push*, pois a informação é enviada sem a requisição do utilizador.
- Emergência: possibilidade de uma pessoa saber a sua localização exata aquando desconhecida.
- Serviços Informativos: possibilidade de oferecer ao utilizador informação baseada na sua localização, como por exemplo, quando o utilizador pretende saber o restaurante ou o cinema mais próximo.
- Navegação: possibilidade de obter direções que o utilizador necessite a partir da sua localização para outra, como é o exemplo do *Google Maps*.
- Lazer: existem jogos que utilizam as posições dos seus utilizadores no jogo, e o caso de grande êxito mais recente, é o do *Pokemón Go*. Para além disso, também existem aplicações que contêm funcionalidades que utilizam a localização dos utilizadores, como é exemplo, das aplicações de desporto que permitem ao utilizador obter os seus dados desportivos, como distância, duração e, até, calorias queimadas.
- Rastrear: hoje em dia já é possível rastrear, em tempo real, os filhos, pacientes com algum problema, o empregador rastrear o seu empregado e os prisioneiros com o uso da bracelete eletrónico de vigilância.

Capítulo 3

Streaming

Ao longo dos últimos anos houve mudanças significativas na forma de como fazer transmissão de vídeos através da Internet. Isto, também, devido ao facto de os utilizadores exigirem, cada vez mais, uma elevada qualidade de vídeo, tempo de inicialização e uma boa reação à interação do utilizador.

Antigamente, quando um utilizador fazia um pedido para visualizar um vídeo, tinha que esperar que o vídeo fosse transferido completamente para o poder visualizar.

Mais recentemente, surgiu o *Progressive Download* baseado no protocolo HTTP onde já era possível visualizar o vídeo sem ter sido feito o download completo do mesmo. Porém o ficheiro era armazenado temporariamente no cliente e o download era feito de uma forma sequencial, ou seja, não seria possível avançar [Swa13].

Hoje em dia, o vídeo já é dividido em *chunk's*, ou seja, o vídeo é dividido em segmentos o que permite ao utilizador ver o vídeo ao mesmo tempo que está a ser descarregado, prevenindo o desperdício do gasto da banda larga, porque se o utilizador não visualizar o vídeo completo, este só descarrega as partes do vídeo que visualizou.

Sendo assim, hoje em dia, existem duas maneiras de fazer transmitir o vídeo, o chamado "*streaming* normal"e o *adaptive streaming*.

- Na transmissão normal, o vídeo é transferido para o cliente com uma só resolução. Neste caso é usado um protocolo de transporte chamado RTMP (*Real-Time Messaging Protocol* que é um protocolo desenvolvido pela Macromedia (comprada pela Adobe em 2005) para *streaming* de áudio, vídeo e dados sobre a Internet, entre o *Flash Player* e o servidor, sendo que este protocolo usa TCP. De forma a penetrar certas *firewalls*, visto ser bloqueado em várias redes, utiliza o protocolo *RTMPT* que encapsula pacotes de dados em pedidos HTTP.

- Enquanto na *adaptive streaming*, que é baseado no protocolo HTTP, o vídeo é codificado para múltiplas velocidades onde um ficheiro manifesto em formato XML é criado no lado do cliente onde é mencionado o nome do *chunk*, o local onde está armazenado e quais os tipos de velocidades disponíveis [TG12]. E, automaticamente, as velocidades de download varia consoante a velocidade da Internet do cliente. O que é uma grande vantagem em relação

à transmissão normal, permitindo uma grande flexibilidade de mudar a qualidade do vídeo instantaneamente [Swa13].

Adaptive HTTP streaming possui várias vantagens, entre elas estão:

1. utilização dos HTTP *caches* e do uso dos *content delivery network* (CDN). Basicamente CDN é uma rede de distribuição de informação que permite fornecer conteúdos de uma forma mais rápida a um grande número de utilizadores. Sendo isto possível, pelo facto, do CDN consistir na distribuição do conteúdo por múltiplos servidores de forma a efetuar a duplicação do mesmo e direcionar o conteúdo ao utilizador com base na proximidade do servidor [SLK12];
2. entrega do conteúdo é dinamicamente adaptável;
3. permite aos visualizadores um começo de visualização rápida e avanço no tempo do vídeo;
4. o cliente pode controlar a troca do *bitrate* tendo em conta o CPU, banda larga disponível, resolução, *codec* e outras condições locais;
5. Firewalls não dificultam nem impedem a entrega via HTTP;

3.1 Técnicas de *Adaptive Bitrate Streaming*

A técnica usada para uma transmissão de multimédia sobre a rede de computadores é *adaptive bit-rate streaming* que é baseado no protocolo HTTP que consiste em pedidos *request-response* entre o cliente e o servidor. Esta técnica baseia-se em detetar a capacidade da banda larga e do CPU do cliente de forma a ajustar a qualidade da transmissão do vídeo, mais especificamente, o vídeo é codificado em várias velocidades e cada uma dessas velocidades são divididas em vários fragmentos, onde a partir do ficheiro enviado no inicio da sessão do servidor para o cliente um *Media Presentation Description* (MPD) que é um manifesto em XML, o cliente têm conhecimento dos fragmentos e das velocidades existentes de cada fragmento. Quando o cliente faz o primeiro pedido, é feito para a velocidade mais baixa, contudo se verificar que a velocidade de download é superior à velocidade de download do fragmento, o próximo fragmento pedido será de uma velocidade superior e vice-versa.

De maneira de disponibilizar a melhor qualidade ao cliente, o *Adaptive Bitrate Streaming* consiste em dividir o vídeo em pequenas partes e codificar em diferentes banda largas, normalmente em baixa, média ou alta qualidade. Estes segmentos são armazenados num servidor HTTP e quando o vídeo começa a ser carregado por parte do cliente ele já faz download da melhor qualidade que é suportado pelo cliente.

HLS, HDS, MMS e DASH são quatro implementações de *adaptive bit-rate streaming* que foram desenvolvidas pela Apple, Microsoft e Adobe, respetivamente. Todas elas se baseiam na mesma ideia e são muito parecidas. Contudo cada uma delas têm as suas próprias características e o seu próprio formato do ficheiro *manifest*.

3.1.1 *Apple HTTP Live Streaming* (HLS)

HLS utiliza ficheiros MPEG-2 TS que encapsula o vídeo e o áudio. HLS define dois tipos de ficheiros, onde a informação é apresentada como comentários ("#"):

- *Normal Playlist* é constituído por URL's que apontam para os segmentos que devem ser tocados sequencialmente. Neste ficheiro, a informação inclui um sequência de números que são associados aos diferentes segmentos, informação sobre a duração do segmento, um sinal que permite saber se os segmentos podem ficar em *cache*, a localização das chaves de descriptografia, o tipo de vídeo e informação do tempo. De maneira a que o cliente de HLS conheça o URL dos mais recentes segmentos disponíveis, é necessário que o ficheiro da *playlist* seja descarregado repetidamente. Isso acontece todas as vezes que um segmento é tocado e, de forma, a minimizar o número de pedidos, é recomendado que esses segmentos tenham duração de 10 segundos, que pode ser considerado longo.

- *Variant Playlist* aponta para uma coleção de diferentes ficheiros normais playlist. Neste caso, a informação inclui o *bitrate* do perfil, sua resolução, *codec* e o ID que pode ser associado a diferentes codificações do mesmo conteúdo.

Na figura seguinte 3.1 podemos observar dois exemplos de ficheiros *manifest* HLS. No ficheiro do lado esquerdo é uma *variant playlist* que mostra oito diferentes saídas com diferentes *bitrates*. Enquanto que, no ficheiro do lado direito, é apresentado um ficheiro *manifest live* que mostra os últimos três segmentos disponíveis.

```
#EXTM3U
#EXT-X-STREAM-INF:PROGRAM-ID=1,BANDWIDTH=531475
mystic_S1/mnf.m3u8
#EXT-X-STREAM-INF:PROGRAM-ID=1,BANDWIDTH=381481
mystic_S2/mnf.m3u8
#EXT-X-STREAM-INF:PROGRAM-ID=1,BANDWIDTH=531461
mystic_S3/mnf.m3u8
#EXT-X-STREAM-INF:PROGRAM-ID=1,BANDWIDTH=781452
mystic_S4/mnf.m3u8
#EXT-X-STREAM-INF:PROGRAM-ID=1,BANDWIDTH=1031452
mystic_S5/mnf.m3u8
#EXT-X-STREAM-INF:PROGRAM-ID=1,BANDWIDTH=1281452
mystic_S6/mnf.m3u8
#EXT-X-STREAM-INF:PROGRAM-ID=1,BANDWIDTH=1531464
mystic_S7/mnf.m3u8
#EXT-X-STREAM-INF:PROGRAM-ID=1,BANDWIDTH=3031464
mystic_S8/mnf.m3u8
```

```
#EXTM3U
#EXT-X-KEY:METHOD=NONE
#EXT-X-TARGETDURATION:10
#EXT-X-MEDIA-SEQUENCE:494

#EXT-X-KEY:METHOD=NONE
#EXTINF:10,505.ts
505.ts
#EXTINF:10,506.ts
506.ts
#EXTINF:10,507.ts
507.ts
```

Figura 3.1: Manifest HLS [YF14]

A principal vantagem é o facto de este protocolo ser simples e facilmente modificado porém existe uma desvantagem muito importante que é o facto e o HLS não ser suportado em plataformas Windows [YF14].

3.1.2 *Adobe HTTP Dynamic Streaming* (HDS)

No HDS, o ficheiro *manifest* contém informação sobre os segmentos disponíveis. Como acontece no HLS, o cliente do HDS descarrega repetidamente dados que lhe permitem saber quais os URLs com os segmentos disponíveis, isto é chamado, *bootstrap information*, sendo que esta informação encontra-se no formato binário.

A grande vantagem do HDS é pelo facto que o *Flash* estar disponíveis em múltiplos dispositivos e está instalado na maior parte dos computadores [YF14].

3.1.3 *Microsoft Smooth Streaming* (MSS)

Neste protocolo, o ficheiro *manifest* contém informação sobre o *bitrate* e resoluções dos segmentos disponíveis, onde o vídeo e o áudio são apresentados separadamente.

As vantagens do MSS inclui na criação de um formato agregado para a *stream*, ou seja, pequenos número de ficheiros podem conter toda a informação sobre o vídeo. A grande desvantagem é que nos computadores, MSS requisita a instalação de um plug-in *Silverlight* [YF14].

3.1.4 *Dynamic Adaptive HTTP Streaming* (DASH)

DASH é o protocolo mais completo e mais complexo entre todos eles, podendo usar fragmentos TS ou MP4. Na figura 3.2 podemos observar a forma de como funciona o MPEG-DASH como foi mencionado anteriormente.

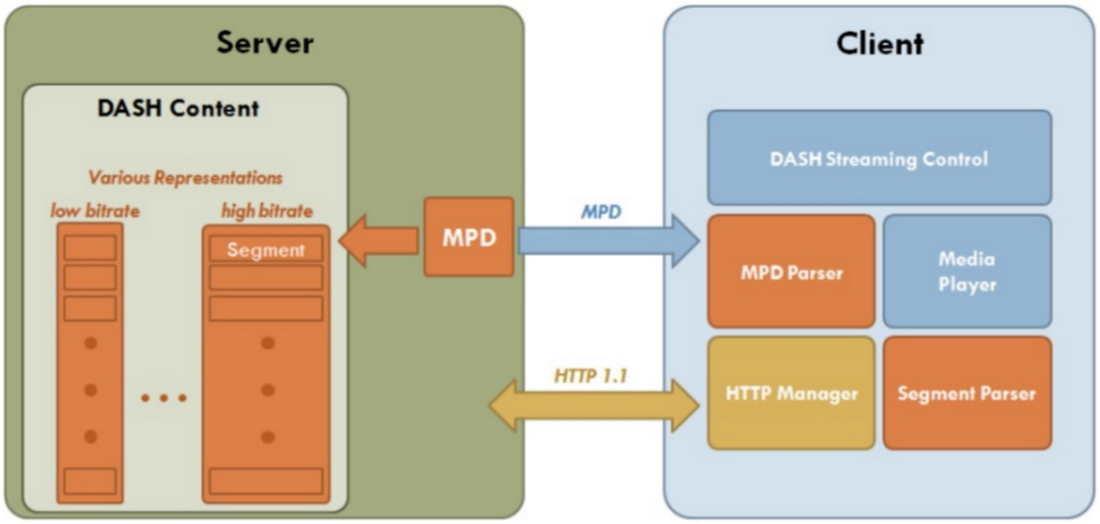

Figura 3.2: Arquitetura MPEG-DASH [1]

A grande vantagem do DASH é ser *open source*. Porém a grande desvantagem é o facto de diferentes implementações DASH serem incompatíveis.

[1] Adaptação da imagem em https://pt.slideshare.net/slederer/mpegdash-open-source-tools-and-cloud-services

Um MPD é uma estrutura de dados sobre os conteúdos de um vídeo. Estes dados são acessíveis ao cliente de forma a fornecer o vídeo ao mesmo.

- Um MPD é constituído por uma sequência de um ou mais Períodos. Um Período permite mudar um conjunto de informações, como a localização do servidor e os parâmetros de codificação. Este conceito foi introduzido de forma a dividir os conteúdos, como os anúncios e o conteúdo do segmento. A cada Período é lhe atribuído um tempo inicial.
- Cada Período contêm uma ou mais Representações do mesmo conteúdo. A Representação é uma escolha possível do conteúdo do vídeo que normalmente são diferenciados através do *bitrate*, resolução e *codecs*.
- Cada Representação consiste em um ou mais segmentos, contendo, no máximo, em um segmento de inicialização e por um ou mais segmentos do vídeo. Este segmento de inicialização contem informação para aceder à Representação mas não sobre dados do vídeo.
- Segmentos contém dados sobre o vídeo.

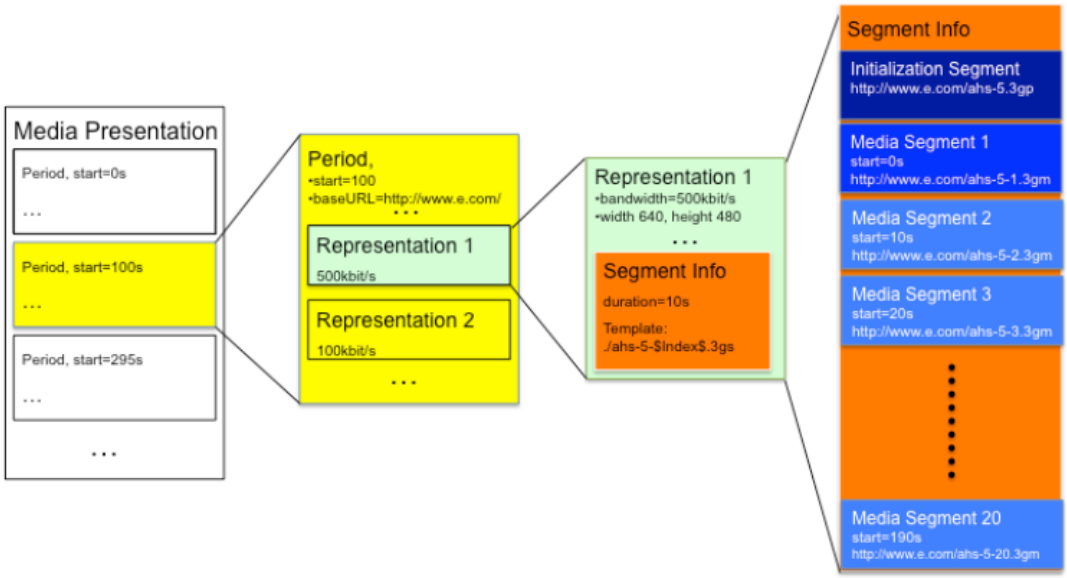

Figura 3.3: Modelo MPD [Sto11]

3.1.4.1 A camada de Transporte

O protocolo de transporte que o *MPEG-DASH* utiliza é o *Transmission Control Protocol* (TCP). Este protocolo entrega os pacotes de dados na sequência correta e assegura a verificação de erros. A ligação entre o emissor e o recetor começa com *handshake* de três vias que permite assegurar uma conexão entre o emissor e o recetor. Durante a transferência dos dados, o recetor

vai confirmando a receção dos pacotes de dados enviando mensagem *ACKnowledgment* e cada pacote é registado com um número sequencial. Nesta ligação existe o uso de uma técnica chamada *checksum* que permite detetar falhas em segmentos específicos.

Outro protocolo de transporte é o USER DATAGRAM PROTOCOL (UDP), no entanto, neste protocolo não há uma garantia em que os pacotes de dados são entregue. O UDP entrega os pacotes de dados independentemente, e consequentemente, esta entrega pode ser fora de ordem e até pacotes de dados podem ser perdidos. Sendo então o UDP um protocolo mais simples que O TCP, indicado para aplicações que necessitem de rápidas transferências de dados e, garantindo, uma boa eficiência.

Tabela 3.1: TCP vs UDP

Característica	TCP	UDP
Tamanho do pacote Header	8 bytes	20-60 bytes
Entidade	Segment	Datagram
Conexão Orientada	Sim	Não
Transporte confiavél	Sim	Não
Ordenação na entrega	Sim	Não
Checksum	Sim	Sim

3.1.5 Comparar os quatro formatos de *Adaptive Bitrate Streaming*

Irá ser, agora, comparado e discutido a usabilidade do DASH, HLS, HDS e MSS. Onde a cor verde indica que aquela implementação suporta uma determinada característica, a cor vermelha indica que aquela implementação não suporta essa característica e a cor preta significa que aquela implementação suporta determinada características mas não tão eficiente quanto as outras [YF14].

Feature	HDS	HLS	MSS	MPEG-DASH
Multiple audio channels	✔	✓	✓	✓
Encryption	✓	✗	✓	✓
Subtitles / Closed captions	✓	✓	✓	✓
Efficient Ad Insertion	✔	✔	✔	✓
Fast Channel Switching	✓	✗	✓	✓
Protocol Support's multiple CDNs in parallel	✗	✗	✗	✓
Agnostic to Video/Audio codecs	✗	✗	✗	✓

Figura 3.4: Comparar todos os formatos

- Entrega de múltiplos canais de áudio: HLS, MSS e DASH entregam o vídeo e áudio separadamente, o que é importante em locais com múltiplas línguas. Enquanto no HDS, o vídeo e o áudio estão juntos nos fragmentos.
- Encriptação: HDS suporta encriptação para cada ficheiro TS, ou seja, cada ficheiro TS é encriptado e não pode ser extraído sem a chave. A localização dessa chave é incluído no ficheiro *manifest*. O HLS não suporta qualquer tipo de mecanismo de codificação. O MSS usa uma *framework* chamada *PlayReady* que permite encriptar o conteúdo, gerindo as chaves, entregando os aos clientes.
- Legendas: Todos os formatos suportam legendas, sendo que, normalmente, elas são referenciadas no *manifest* e armazenadas num ficheiro individual.
- Inserção eficiente de anúncios: como o vídeos está repartido em segmentos, basta inserir o anúncio num segmento especifico e substitui lo pelo segmento original. O que permite a continuação da utilização do mesmo servidor HTTP porém será necessário redirecionar o pedido para o segmento especifico que contém os anúncios. MPEG-DASH permite uma interface padrão, os Períodos que permitem a inserção dos anúncios de uma forma eficiente.
- Troca rápida de canal: esta característica está relacionada com o tamanho dos segmentos, pequenos segmentos permitem uma troca mais rápida do que longos segmentos. HLS tipicamente usa segmentos de 10 segundo, enquanto que os restantes usam entre 2 a 4 segundos.

- Múltiplo suporte de CDNs em paralelo: só o MPEG-DASH oferece um mecanismo através dos *Base URLs* no *manifest*, onde múltiplas CDN são usadas. O cliente é, então, capaz de selecionar a CDN que melhor performance lhe fornece.

- MPEG-DASH é agnósticos tanto a *codecs* de vídeo como de áudio. Em vídeo podem ser usados vários como: H.264/AVC, H.265/HEVC, MPEG-2 Video, VP8, VP9, entre outros, enquanto que em áudio pode ser usado, por exemplo, MP3 e AAC.

3.2 Codificação vídeo

A área de compressão de imagem e vídeo tem crescido rapidamente nos últimos anos. O principal objetivo da codificação de imagem e vídeo é alcançar um baixo *bit rate* (números de bits processados por unidade de tempo) de armazenamento de dados e de transmissão, mantendo uma deformação aceitável. Hoje em dia, mais de 2.6 milhões de horas de vídeo são carregadas no Youtube a cada mês e, também cada vez mais, com uma melhor resolução. Contudo, estes dados são gigantes que necessitam de enormes recursos para armazenamento e transmissão [ZAD+00]. A compressão pode ser classificada em duas categorias:

- *lossless*: os dados descodificado são exatamente os mesmos que os originais. Contudo, o *entropy coding* não é suficiente para alcançar a taxa de compressão desejada.

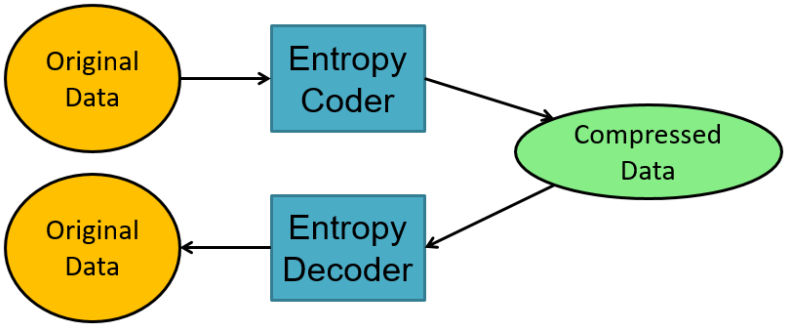

Figura 3.5: Lossless

- *lossy*: o objetivo da televisão é transmitir um vídeo para o recetor. Desde que o recetor é um espetador, o objetivo não é, necessariamente, transmitir a representação exata da realidade mas sim algo que o utilizador perceba como parecido à realidade. Como nós, humanos, temos um limite na capacidade de perceber detalhes, este tipo de compressão toma isso como vantagem dessa limitação, removendo:

  1. redundância: está relacionado com similaridades.
  2. irrelevância: está relacionado com informação impercetível ao olho humano, ou seja, certos detalhes desnecessários são removidos.

O problema desta compressão é fazer uma taxa tão baixa quanto possível, mantendo a mesma qualidade visual pois os dados descodificados e a imagem original não são os mesmo.

3.3 Meta-Informação

3.3.1 VAST

VAST que advém de *Video AD Serving Template* é responsável por fornecer informação aos leitores de vídeos sob um formato XML sobre qual anúncio usar, como o anúncio será exposto, se os utilizadores poderão avançar ou não o anúncio e quanto tempo demorará esse anúncio. Além disso este *template* permitiu ao leitor de vídeo e ao servidor de publicidade comunicaram sempre da mesma maneira.

Antes do aparecimento do VAST era dispendioso fazer a comunicação entre o leitor de vídeos e o servidor de publicidade pois cada leitor de vídeo é desenvolvido numa tecnologia diferente, o que levaria ao servidor de anúncios enviar a informação do anúncio de maneira diferente para cada tipo de leitor de vídeos. Logo VAST permitiu aos leitores de vídeos mostrarem anúncios de qualquer servidor de anúncios desde que esse mesmo leitor de vídeo consiga pedir e processar de um documento XML.

Outra ponto importante é perceber o funcionamento da comunicação entre o leitor de vídeos e o servidor de anúncios. É um processo simples que pode ser explicado em três passos: em primeiro lugar o leitor de vídeos faz um pedido VAST ao servidor de anúncios, e de seguida o servidor de anúncios faz uma resposta VAST que contém todos os caminhos para ir buscar e mostrar o anúncio [LCS+12].

3.3.2 VMAP e VAST

VMAP (*Video Multiple Ad Playlist*) também sob um formato XML permite a quem produz o conteúdo do vídeo gerir o controlo dos anúncios utilizados, possibilitando que este defina o tempo para cada anúncio, quantos anúncios são permitidos e que tipo de anúncios.

Podemos assim perceber que o VMAP não funciona sem o VAST porém o VAST funciona sem o VMAP, pois o VMAP é usado para definir quando ocorrem os anúncios e o VAST é usado para definir qual ou quais os anúncios são inseridos naquela momento definido pelo VMAP.

Portanto o VMAP é complementar ao VAST, onde o funcionamento de ambos interagem da seguinte maneira como indica a figura 3.6: o VMAP representa uma estrutura que define os anúncios no vídeo onde faz um pedido ao servidor e espera uma resposta VMAP que contem um conjunto de anúncios e em cada anúncio é constituída por uma resposta VAST que fornece ao leitor de vídeo anúncios específicos para inserir naquele intervalo especificado pelo VMAP. Ou seja, o leitor de vídeo mostra os anúncios VAST no tempo especificado pelo VMAP [Bur14].

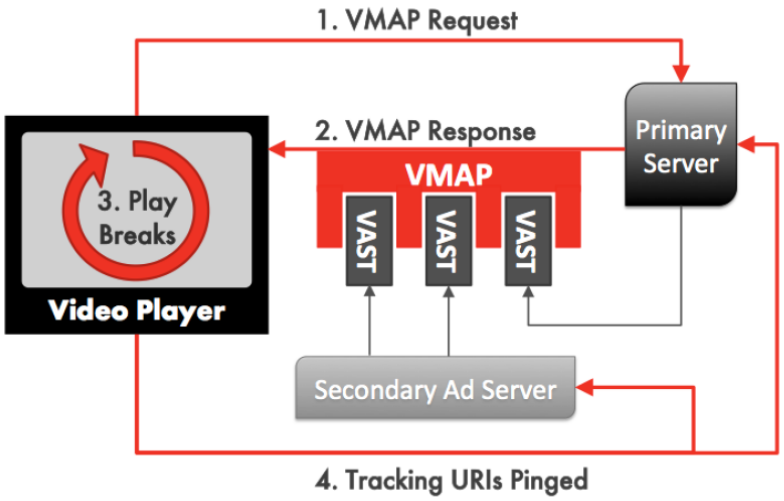

Figura 3.6: Arquitetura VMAP e VAST [Bur14]

3.4 Validação de um XML

XML, abreviatura de *Extensible Markup Language*, é neste momento uma ferramenta que possibilita a representação de dados e o facto de ser bem definido e fácil de usar permiti-lhe essa designação.

Um documento XML deve ser bem formado, ou seja, deve respeitar um conjunto de regras, como por exemplo, um ficheiro XML deve começar com uma declaração XML, um elemento *root* deve ser único, as *tags* XML devem ser sensíveis ás maiúsculos e minúsculas, todos os elementos devem ser fechado. E para isso, existem duas maneiras diferentes que permite especificar a estrutura desses mesmo documentos propostas pelo W3C: *Document Type Definition* (DTD) e *XML Schema* (XSD) [MN09].

Hoje em dia, o XSD é a linguagem XML Schema mais importante, mais utilizada [BNVdB04] e mais poderosa que os DTD's contudo é mais complexo e mais difícil de trabalhar [MN09].

DTD é a linguagem mais antiga das duas e a XML Schema tentou corrigir alguns erros que a DTD tinha. Sendo que, a linguagem XSD tem algumas vantagens em relação ao DTD:

- permite indicar a origem dos elementos do schema.
- é escrito em XML o que permite a que não seja preciso aprender outra linguagem.
- permite definir o tipo de dados dos elementos, e até o seu comprimento especifico ou valores o que permite avaliar se os dados do documento XML são precisos.
- permite definir a ordem dos elementos filhos e o que leva a dizer que XDS fornece um maior controlo na estrutura do XML.

Capítulo 4

Trabalhos Relacionados

As soluções analisadas de seguida têm como foco a personalização de anúncios via *product placement*. Personalizar anúncios e fazer *product placement* são temas correntes e interessantes para a industria de *media*, sendo que um dos principais objetivo é aumentar as receitas advindas da publicidade comparando com a publicidade tradicional.

4.1 Personalização de Objetos em Vídeo

O objetivo deste trabalho é a personalização de vídeos, inserindo objetos neles, de acordo, com o perfil do utilizador. Focando-se, portanto, numa aplicação e personalização de *product placement*.

O trabalho realça dois aspetos: as tecnologias que permitem a integração arbitrária de objetos externos no vídeo e uma plataforma comercial que permite os objetos serem selecionados e integrados no vídeo original de acordo com o perfil do utilizador.

Para compressão de vídeo foi utilizado o MPEG-4. O conteúdo do vídeo é produzido com espaços onde objetos externos podem ser inseridos, esses espaços são definidos por *placeholders* onde estão definidos na meta informação. As propriedades do *placeholders* definem o tipo de objeto que pode ser inserido, sendo que estes objetos são guardados em diferentes bibliotecas. A meta-informação é necessária para descrever os objetos existentes na biblioteca e para descrever as características do local quando o vídeo pede a inserção de um objeto.

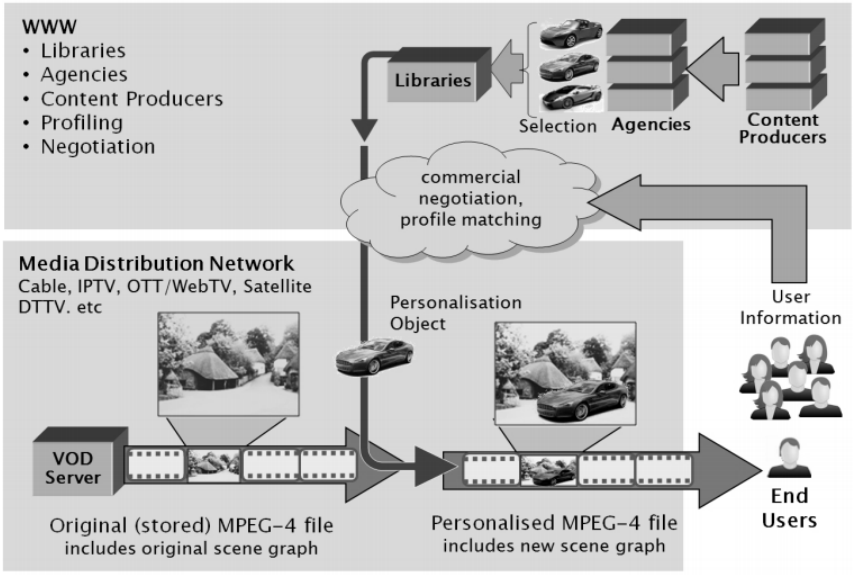

Figura 4.1: Objetos adquiridos por uma biblioteca e inseridos no vídeo [FMB12]

De maneira a criar o perfil do utilizador, além de se usar todos os dados pessoais dele, também, utilizaram-se *tags* que os utilizadores usaram para classificar um vídeo e quanto mais vezes um utilizador utilizar essa *tag* e quanto maior for a classificação dada a esse vídeo mais influencia terá. E para analisar os resultados, é criado um grafo onde os nós correspondem às *tags* no sistema e as arestas representam a relação entre as *tags*, onde esta relação é aumentada sempre que duas *tags* aparecem junto na classificação do mesmo vídeo. Na figura 4.2 podemos observar um exemplo de um grafo gerado a partir de *tags* [FMB12].

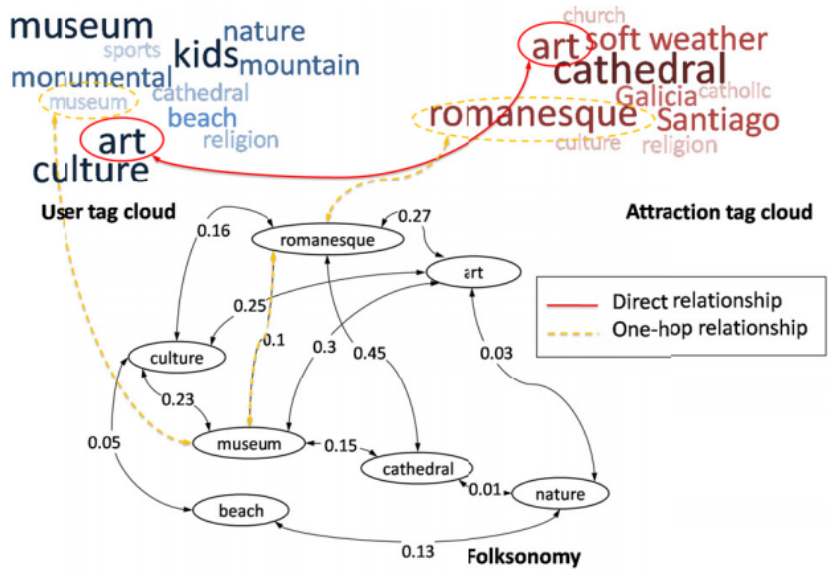

Figura 4.2: Filtro baseado no grafo [FMB12]

Resumindo, esta é uma proposta para personalização do vídeo baseado no perfil do visualizador e no contexto (o programa selecionado, etc) e esta personalização é alcançada através de uma inserção dinâmica de objetos no vídeo.

4.2 Plataforma para Product Placement

Este trabalho propõe uma abordagem que integre personalização de *product placement* envolvendo os anunciantes, distribuidores de conteúdo e visualizadores. Esta personalização de anúncios em vídeos é feita consoante o utilizador final.

Neste cenário, os anunciantes criam e descrevem os produtos para a publicidade, enquanto que os distribuidores fornecem o vídeo com os produtos personalizados aos utilizadores e preocupam-se com a definição do perfil do utilizador.

Por um lado, os anunciantes pretendem colocar os seus anúncios nos vídeos pelo preço mais baixo possível e, por outro, os distribuidores desejam vender os espaços no vídeo para preencher com os objetos que sejam mais apropriados ao perfil do visualizador. Para alcançar esse objetivo, os distribuidores enviam aos anunciantes produtos que combinam com os perfis dos visualizadores para negociar. Esta proposta para personalização de *product placement* integra: uma plataforma que permite negociar os produtos entre os anunciantes e os distribuidores; ter o perfil do visualizador (do lado do distribuidor); *product placement* no vídeo do utilizador (do lado do distribuidor e do visualizador). A localização dos objetos, a nível de tempo e espaço, são definidos num documento que é transmitido com o vídeo e, esse documento, é posteriormente atualizado com os sítios escolhidos e utilizados para no final serem descodificados e interpretados. A escolha do produto a inserir no vídeo depende do perfil do utilizador, do contexto do conteúdo do vídeo e dos objetos disponíveis.

O perfil do utilizador é construido usando técnicas de data *mining* baseado em programas classificados e vistos pelo utilizador [VMBF16].

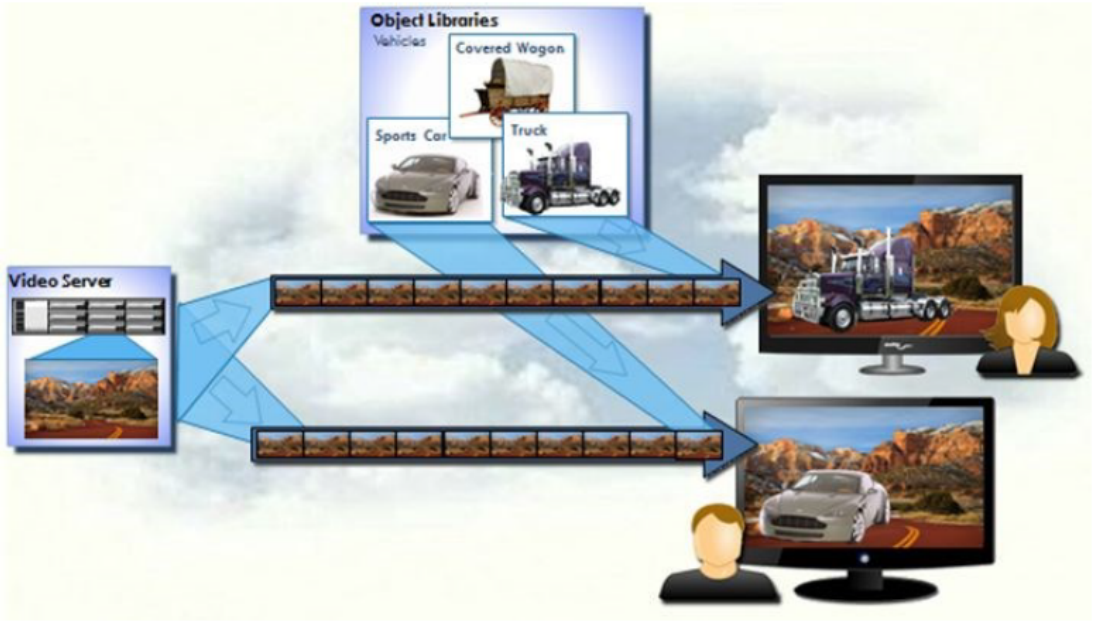

Figura 4.3: *Product Placement [VMBF16]*

4.3 Conclusões sobre os Trabalhos Relacionados

É possível apurar, perante as duas soluções anteriormente apresentados, que o uso de *product placement* já é algo conhecido e utilizado na indústria contudo, a contextualização dos anúncios no vídeo, em ambas as soluções, é baseado no perfil do utilizador.

Pode-se afirmar, que o uso da localização para contextualizar o *product placement* em vídeo é um passo, ainda não dado, que poderá ser mais eficiente. Pois as marcas irão se dirigir ao consumidor na altura certa o que poderá aumentar a probabilidade do efeito dessa publicidade perante o consumidor.

Nestas soluções, não foi nunca menciona a técnica de transmissão em vídeo, muito menos o MPEG-DASH, podendo também afirmar, que o uso dessa implementação é um passo em frente visto ser mais eficientes e mais vantajosas em relação a outras técnicas.

Capítulo 5

Solução

Indo de encontro da oportunidade na indústria ainda não explorada, pretende-se desenvolver uma aplicação que consista fazer *product placement* em televisão, possibilitando a inserção de segmentos personalizados, ou seja, incorporados com anúncios, consoante a localização do utilizador final.

Na prática, o objetivo final será que, a partir do mesmo vídeo, diferentes utilizadores, de diferentes localizações, visualizem diferentes anúncios. Na imagem 5.1 podemos ver um exemplo, onde as mesmas imagens têm diferentes objetos nelas inseridas.

Figura 5.1: Anúncios no mesmo vídeo para utilizadores de diferentes locais

5.1 Arquitetura

Para fazer *product placement* em televisão é proposto uma aplicação que permita a troca entre o vídeo original e os anúncios contextualizados consoante a localização do utilizador e, através de uma *Web App* permitir tocar os segmentos do vídeo. Para atingir esse objetivo, apresentamos a seguinte arquitetura geral 5.2. Nos próximos diagramas apresentados, relacionados com a arquitetura, todos os componentes com cor de Bege são componentes que já se encontravam desenvolvidos e implementados pela *MOG Technologies*.

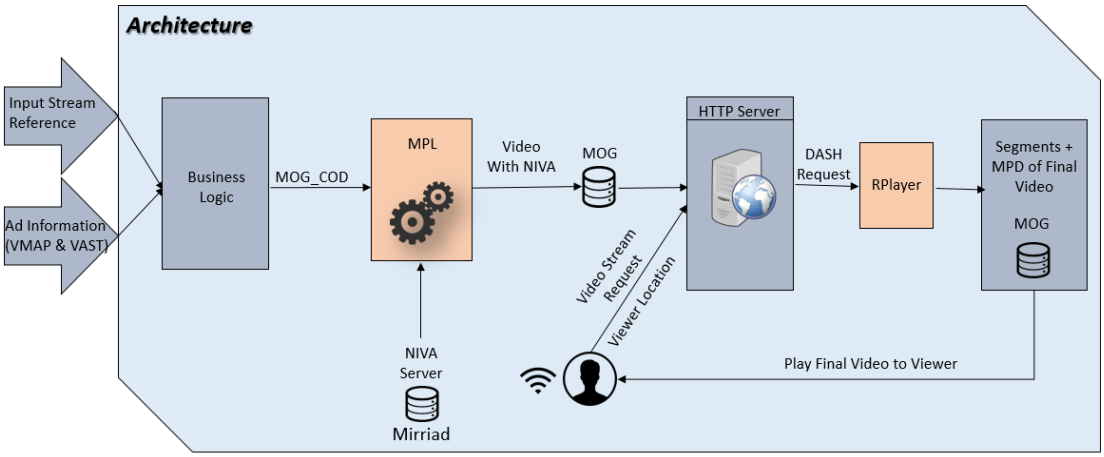

Figura 5.2: Arquitetura

De uma maneira muito sucinta, a aplicação recebe a informação sobre os anúncios, criando uma estrutura, aceite, pela biblioteca da MOG, o MPL, no qual irá construir o vídeo original com os anúncios NIVA, que estarão armazenados numa base de dados da Mirriad. De seguida, esse vídeo já com os NIVA inseridos, será armazenado numa base de dados da MOG. Quando houver um pedido do cliente para visualizar um vídeo, será imediatamente detetado a sua localização e, dependendo da mesma, será indicado, ao RPlayer, qual o vídeo a fazer DASH, de maneira a criar DASH e armazenando-os, outra vez, numa base de dados da MOG. Por fim, disponibiliza-os ao cliente.

Para um melhor entendimento da arquitetura, a solução é composta por duas partes. O esquema da figura 5.3 demonstra a arquitetura da primeiro parte da proposta, onde se pode verificar, que a mesma é composta por dois módulos principais, cada uma com uma função especifica e distinta. Será possível entender como será efetuada a troca entre o vídeo original e os anúncios de acordo com a localização do utilizador final.

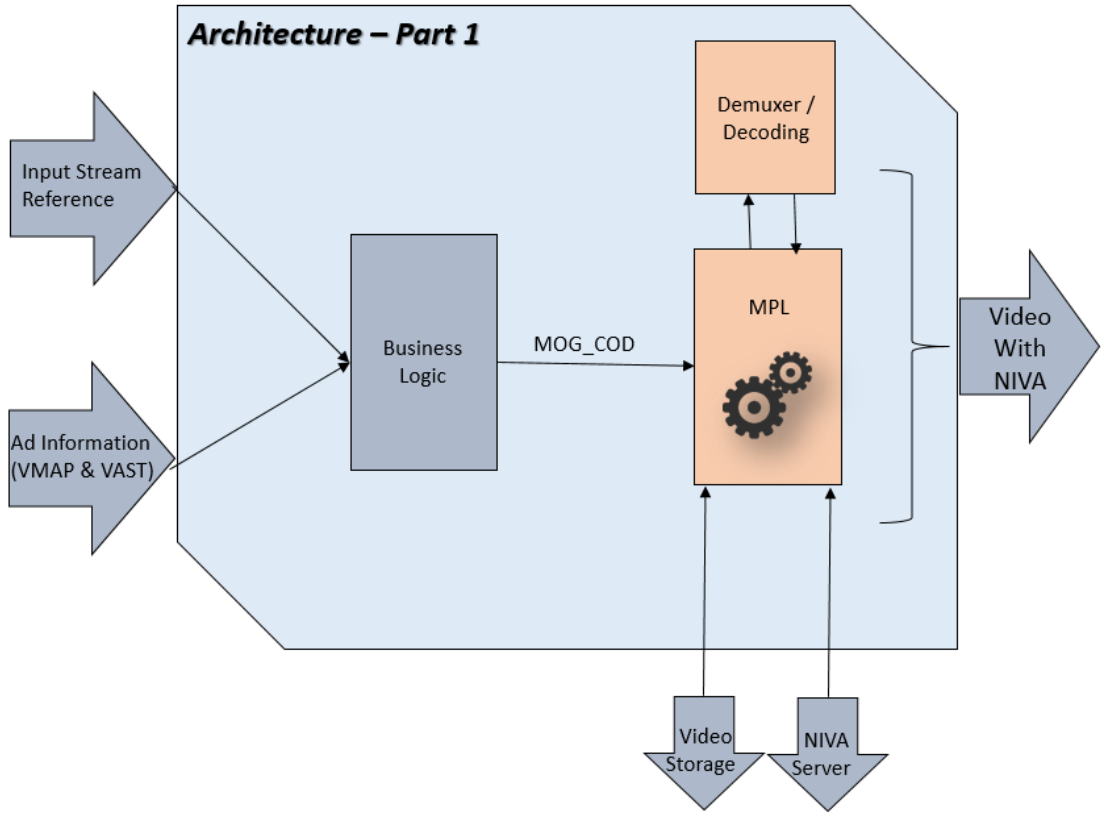

Figura 5.3: Arquitetura - Parte 1 - da solução proposta

Os módulos que compõem a primeira parte da arquitetura proposta são:

- *Business Logic* recebe como entrada informação sobre os anúncios (VAST e VMAP) e sobre o vídeo original. Este módulo é responsável por processar esses mesmo XML's e por os validar contra os seus *XML Schemas*. Também é encarregue por gerar uma `MOG_COD` (*MOG_Common_Object_Descriptor*), a partir, da informação recolhida do VAST e VMAP.
- *MPL* é a biblioteca da MOG Technologies, em que fica responsável por receber uma `MOG_COD`, vinda do *Business Logic*, e a partir, da mesma irá resultar o vídeo final incorporados com os anúncios NIVA.

A *Mirriad*, empresa especializada em técnicas por visão de computador, será a responsável por inserir os anúncios nos vídeos. Onde já existirá um pré processamento por parte deles, e que irão disponibilizar, no *NIVA Server*, um conjunto de segmentos iguais do programa porém com diferentes objetos neles inseridos [Mir].

Native Advertisement é computacionalmente muito exigente para ser processado em tempo real, por isso, esta proposta é constituída por múltiplas sequencias do programa, cada uma com objetos diferentes objetos inseridos neles. Por esse motivo, não se irá tratar de um processo em tempo real porque para o realismo, é necessário considerar sombras, reflexões, entre outros, dos quais precisam de ser pré-processados.

Em suma, o servidor dos anúncios irá armazenar segmentos do programa em múltiplas versões, onde o mesmo pedaço do programa tem diferentes produtos neles inseridos em cada versão.

É apresentada a segunda parte da arquitetura no diagrama seguinte 5.4. Esta parte é o que chamamos *Dash Generator* em que se recebe como entrada um pedido de um visualizador para visualizar um vídeo e, como o próprio nome indica, este módulo é responsável por gerar diferentes representações dos segmentos e fornece-los. Para isso, existe uma comunicação com uma ferramenta interna, *RPlayer*, através de uma API REST.

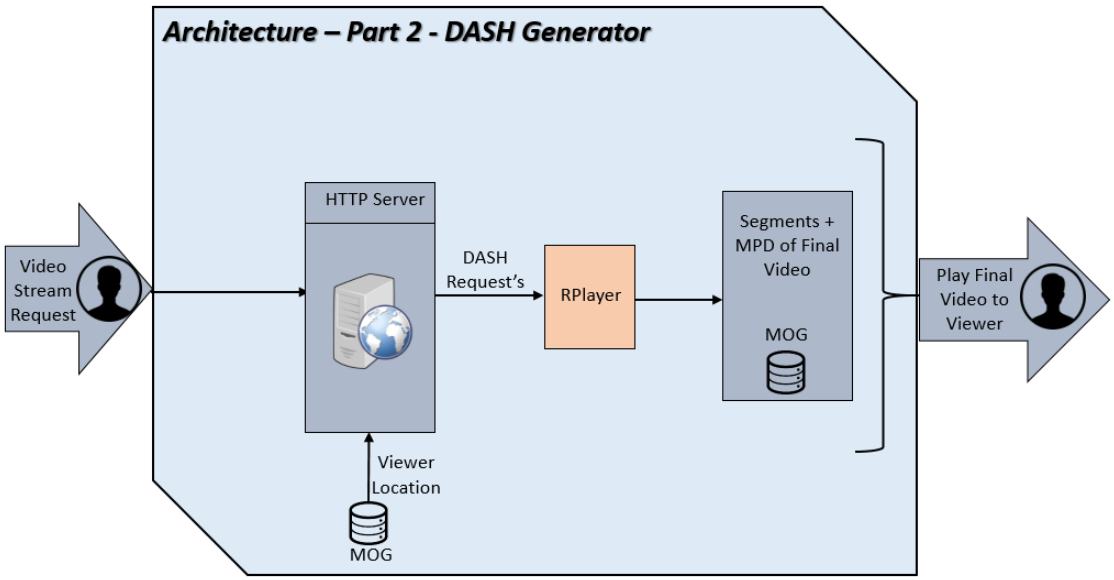

Figura 5.4: Arquitetura - Parte 2 - da solução proposta

5.1.1 Expansibilidade e Modularidade

Cada módulo na aplicação é independentes das restantes. Esta modularidade da aplicação permite que sejam criados novos tipos de componentes sem interferir com as restantes. As possibilidades de expansibilidade permitem respostas a diferentes necessidades e criação de *workflows*. Poderíamos adicionar, facilmente, por exemplo, um `MOG_COD` *Storage*.

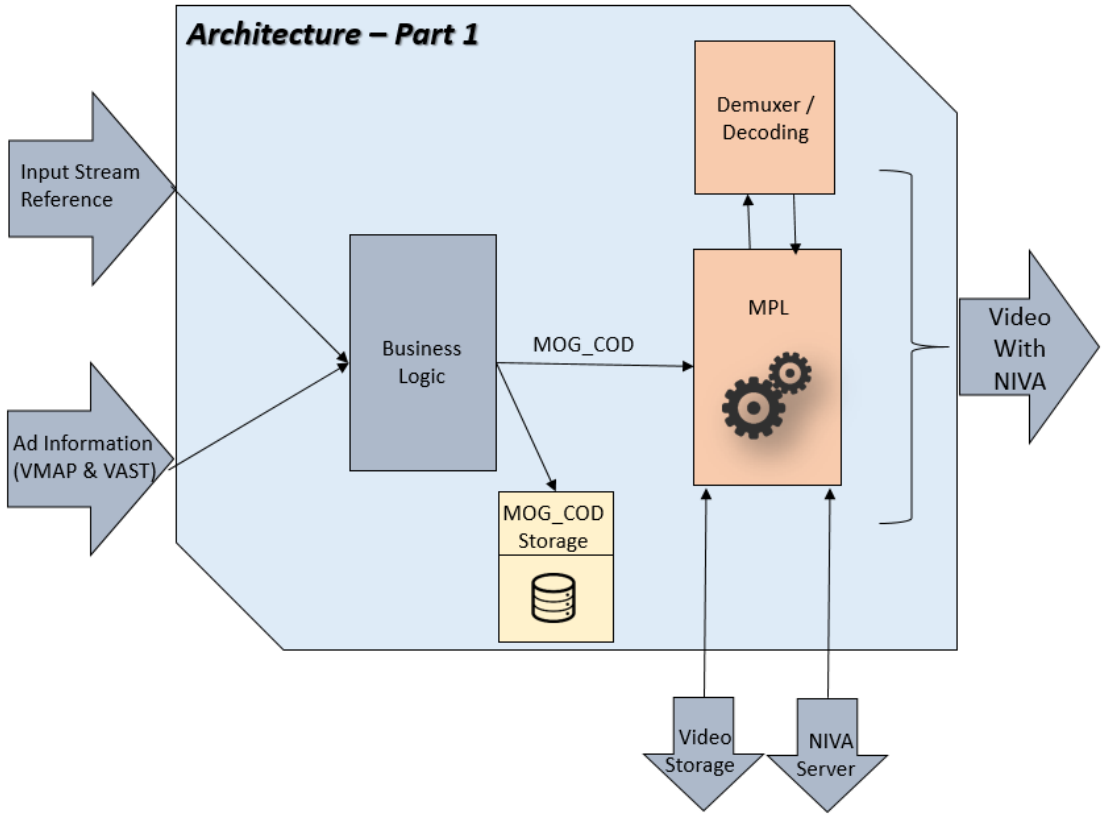

Figura 5.5: Exemplo de expansibilidade da arquitetura

5.2 *Workflow*

É importante compreender, com mais detalhe, como os módulos da arquitetura se interagem.

1. *Business Logic* processa e valida o VMAP e VAST usando a biblioteca MSXML.
2. A partir dos dados do VMAP e VAST é criado uma `MOG_COD`.
3. Através da `MOG_COD`, o *Business Logic* informa o MPL sobre o número de segmentos que é preciso mudar, o tempo exato que é preciso inserir esses segmentos que contém a publicidade, bem como a sua duração e localização.
4. MPL é responsável por receber informação do *Business Logic* e por trocar entre o vídeo original e os segmentos mais apropriados. MPL vai buscar os segmentos para trocar ao *NIVA Server*. A nossa abordagem é ter uma biblioteca com múltiplos segmentos do programa, cada um com um diferente objeto inserido nele. Estes serão guardados no *NIVA Server* e a localização do utilizador irá identificar qual será o mais indicado.
5. *Demuxer*, é um componente, que já se encontra desenvolvido pela MOG para diversos *containers*, nomeadamente, MXF, MOV, MP4 e TS, que pertence ao MPL no qual recebe os

vídeos. Sendo responsável por dividir o ficheiro de vídeo em elementos individuais tais como: vídeo e áudio, e posteriormente, envia-los para os respetivos *descodificadores*. Um *codec*, que significa codificar/descodificar, de vídeo é o que permite comprimir e descomprimir o vídeo sem grandes perdas visíveis ao olho humano. Para isso, foi usado vários tipos de *codecs* como o da Sony, PanaSonic e Main Concept. De seguida é possível fazer operações sobre esses dados, como por exemplo, em vídeo (alterar o tamanho ou alterar a conversão do espaço de cor do mesmo) e no áudio (converter para mono) e quando, finalizado todas as operações, volta-se a codificar todos os dados.

Na seguinte imagem 5.6 é apresentado uma visão mais detalhada sobre *Demuxer* de maneira a que seja possível perceber melhor em que consiste este componente.

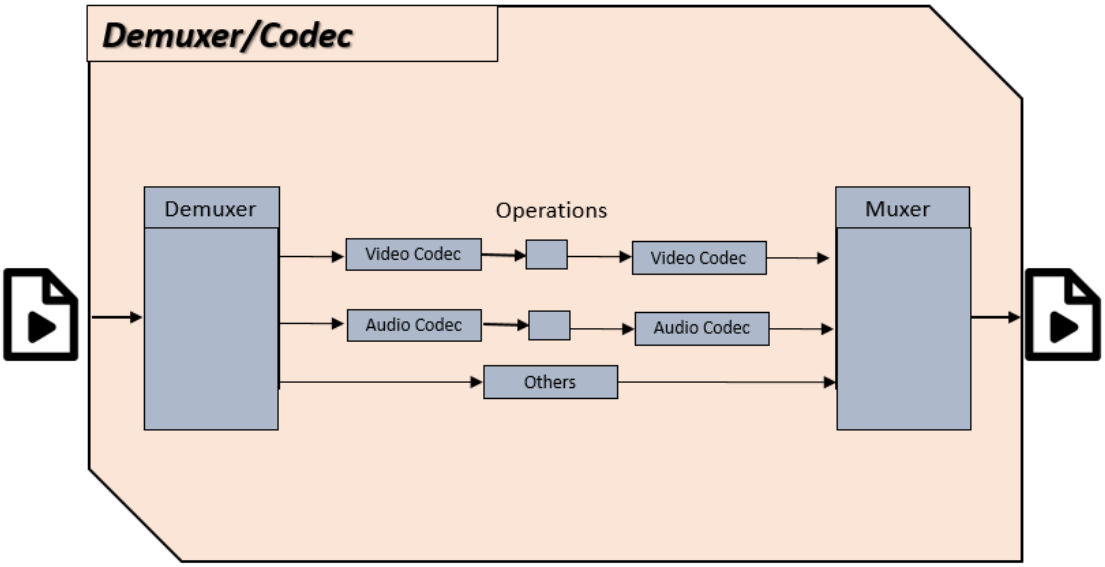

Figura 5.6: Demuxer

6. A partir da informação da `MOG_COD` proveniente do *Business Logic*, o MPL irá ter como saída o vídeo final com o anúncios já inseridos. Esse vídeo será armazenado em recursos da *MOG Technologies*. Resumidamente, o MPL, recebe um `MOG_COD`, e partir dessa informação, ocorre uma operação no MPL à qual chamamos *Ingest*, operação esta, que como já mencionado anteriormente, já estava implementada pela *MOG Technologies*, na qual origina o vídeo com NIVA.

7. Depois do servidor receber o pedido para visualizar o vídeo por parte do cliente, o servidor deteta a localização do cliente através do seu IP. E, a partir, dessa localização, o servidor irá escolher o vídeo a apresentar ao cliente.

8. O servidor depois de ter conhecimento do vídeo a mostrar ao cliente, envia pedidos via REST ao *RPlayer*, criando segmentos de DASH, de 2 segundos, armazenando-os e fornecendo-os.

5.3 Implementação

Depois de perceber a arquitetura e o seu *workflow*, iremos tentar entender mais ao pormenor o que cada módulo faz em concreto e como.

5.3.1 Business Logic

No seguinte diagrama 5.7 é apresentado uma visão mais detalhada sobre o módulo *Business Logic* de maneira a que seja possível perceber melhor em que consiste este módulo.

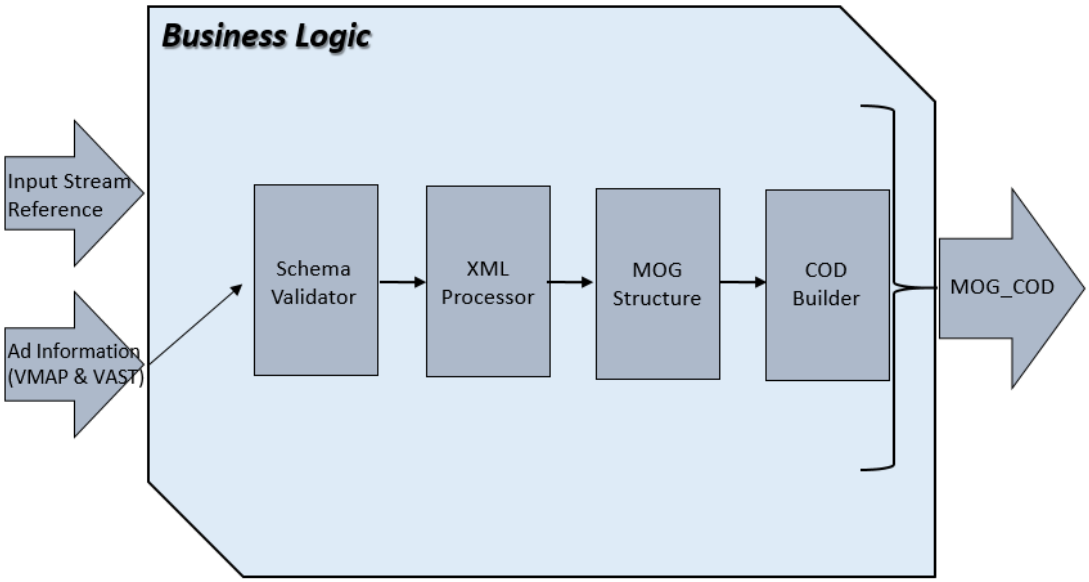

Figura 5.7: Business Logic

A *MOG Technologies* estava há procurar de uma solução onde pudesse receber os dados sobre os anúncios. Depois de se chegar à conclusão que o uso do VAST e VMAP seria uma boa opção para todos os intervenientes (*MOG Technologies e Mirriad*), houve um estudo sobre toda a informação que esses ficheiros continham e quais seriam os dados que interessavam para a *MOG Technologies*.

Depois de receber o VMAP e o VAST, que são dois XML's, passam por um processo de validação à qual designamos *Schema Validator*. Posteriormente, a essa verificação, asseguramos que a aplicação recebe esses XML's sem qualquer tipo de problemas, tanto a nível de *tags* como a nível de tipos, avançamos para a próxima fase que consiste no processamento dessa informação. Essa informação depois de processada será toda organizada e armazenada numa estrutura da *MOG*

Technologies. Após esses dados serem armazenados numa estrutura da MOG é construída um `MOG_COD` que irá servir como entrado para o próximo módulo, contendo uma `MOG_EDL`.

No VMAP, pode-se observar um exemplo de um *Ad Break* num ficheiro VMAP. A informação mais importante é o *time offset* porque é esse valor que irá indicar o tempo exato que o segmento precisa de ser inserido no conteúdo do vídeo. Um outro importante facto é que recebemos uma resposta VAST, onde podemos observar um exemplo do mesmo em VAST, na qual esta tem mais informação sobre o anúncio, como a sua duração e localização.

MOG_Edit_Decision_List é usada em processos de pós-produção de edição de vídeos. EDL é uma estrutura de dados que especifica a configuração temporal dos segmentos. Referindo o tempo inicial e a duração de cada segmento [And97]. No anexo `MOG_COD` encontra-se um exemplo de uma `MOG_COD`. Pode-se constatar que o resultado desta operação é um ficheiro. Consegue-se verificar que, neste exemplo, o vídeo final é um conjunto de 3 *clip's*. Em que, por cada *clip*, é referido o tipo de ficheiro de vídeo e o seu local. Além de que é inserido o tempo inicial e o tempo final que queremos que aquele *clip* toque. Neste exemplo em particular, pode-se reparar que inicialmente, irá ser utilizado os primeiros 8 segundos do vídeo original designado por "OportonityCity.MXF", depois desses 8 segundos, será inserido um vídeo que contém o anúncio chamado "AdMacDonals.MXF"durante os próximos 3 segundos, e por fim, voltará ao vídeo original dos 11 segundos aos 15 segundos.

Para complementar as boas práticas de programação, também foram desenvolvidos um conjunto de testes unitários com o principal objetivo de encontrar qualquer tipo de erro.

5.3.2 DASH Generator

Este módulo têm, também, um papel muito importante na nossa solução. Depois do vídeo final estar construído, com os anúncios NIVA nele inseridos, que estão armazenados numa base de dados da MOG, foi desenvolvido um servidor em *Python*, no qual, o principal objetivo é reproduzir o vídeo, em formato DASH, dependendo da localização do utilizador. Foi utilizado uma *framework* chamada *Flask*. O visualizador começa por fazer um pedido para visualizar um vídeo, o servidor começa por detetar a localização do utilizador, através do seu IP e, de seguida, verifica se o vídeo para aquela localização já se encontra em pequenos segmentos:

1. Sim: Disponibiliza ao utilizador o `MOG_MPD` daquele vídeo, onde irá resultar numa troca de pedidos HTTP entre o servidor e o cliente.

2. Não: O servidor, a partir de uma API REST, comunica com uma ferramenta da MOG chamada RPlayer. No qual informa o RPlayer para fazer DASH de um vídeo em específico, depois de o fazer, transfere esses segmentos e o respetivo `MOG_MPD` para um armazenamento da MOG, que inicialmente estavam numa pasta temporária. De seguida, enviará para o cliente o `MOG_MPD`. O uso desta ferramenta, RPlayer, é importante na medida que, deste modo, usaremos uma estrutura de MPD suportada e criada pela MOG.

Será apresentado o *workflow* deste sistema 5.9 de maneira a um melhor entendimento do mesmo.

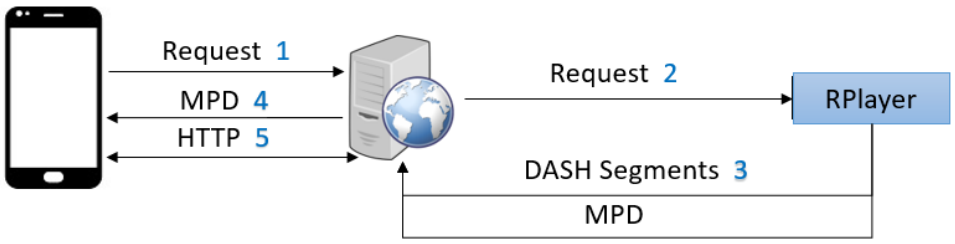

Figura 5.8: RPlayer

1. O cliente faz um pedido ao servidor para visualizar o vídeo.

2. O servidor ao receber esse pedido deteta a localização do utilizador através do seu IP e faz, de seguida, pedidos ao RPlayer de maneira a criar segmentos DASH e respetivo MPD. É neste processo que é definido qual o vídeo a mostrar ao utilizador consoante a sua localização.

3. O servidor guardará os segmentos DASH e o MPD num armazenamento da *MOG Technologies*.

4. O servidor irá responder ao cliente com esse mesmo MPD e, consequentemente, o cliente saberá onde ir buscar os segmentos.

5. Por fim, existe uma troca de pedidos HTTP entre o cliente e o servidor onde o cliente vai pedindo os segmentos que pretende ao servidor.

5.3.3 Bibliotecas e *Framework*

A proposta da solução apresentada foi desenvolvida com recurso a duas bibliotecas: a biblioteca de processamento de *media* da MOG Technologies e a biblioteca MSXML (*Microsoft XML Core Services*) e uma *Framework*: Flask. Será interessante perceber, rapidamente, qual a principal diferença entre uma biblioteca e uma *framework*. Uma biblioteca é uma coleção de definições de classes, cujo principal objetivo é reutilizar código enquanto que uma *framework* chama o código desenvolvido quando necessário. Resumindo, quando se chama um método a partir de uma biblioteca, estamos no controlo mas numa *framework*, o controlo é invertido, ou seja, a *framework* chama o código.

5.3.3.1 Biblioteca MOG (MPL)

A MOG Technologies possui uma biblioteca de processamento *media*, o MPL (*Media Processor Library*), sendo esta biblioteca de propriedade privada e utilizada em todos os produtos que se encontram no mercado. Esta biblioteca contém métodos que permite gerir os conteúdos *media* e funcionalidades que otimizam a gestão de memória. Resumidamente, permite a que aplicações reproduzirem vídeo e áudio de alta qualidade, suportando uma grande quantidade de formatos. Sendo que tirei partido das vantagens e das funcionalidades do *ingest*, onde a maior parte dos produtos da empresa a utilizam. A utilização desta biblioteca é fundamental pois irá permitir a inserção imediata desta solução com os atuais produtos da empresa.

5.3.3.2 Biblioteca MSXML

O MSXML é uma biblioteca da *Microsoft* que permite desenvolver aplicações baseadas em XML, em C++. Esta biblioteca foi essencial para processar os XML's e para a utilização do XPath pois esta biblioteca permite organizar os dados do XML numa estrutura em árvore. XPath, definido pelo *World Wide Web Consortium* (W3C), é uma linguagem que permite navegar através dos elementos e atributos de um documento XML usando uma sintaxe baseada em caminhos, esses caminhos são utilizados para selecionar os nós ou um conjunto de nós de um XML. Também permite validar os documentos XML usando XSD Schema, XSD permite definir a estrutura e os tipos de dados para os documentos XML.

5.3.3.3 Framework Flask

Flask é uma *framework web* escrita em *Python*, fornecendo um modelo simples para desenvolvimento *web*, permitindo economizar o tempo de desenvolvimentos de aplicações *web*. Esta *framework* permitiu, facilmente, definir rotas usando métodos HTTP (*GET, POST, DELETE, PUT*).

Segue-se, um exemplo, da definição da rota principal onde é chamado um ficheiro HTML.

```
1 @app.route('/')
2 def index():
3     return render_template('index.html')
```

5.3.3.4 RPlayer

Para gerar Dash foi utilizada uma ferramenta da MOG Technologies, o RPlayer (*Retocatable Player)*.

RPlayer é um serviço que permite tocar vídeos e deve ser controlado por comandos *REST*. Como já mencionado, este serviço permite a utilização do MPEG-DASH, em que basicamente cria pequenos segmentos e um *manifest* a partir de um ficheiro de vídeo. Esses segmentos de vídeo são codificados em H.264, o áudio em AAC e ambos são encapsulados em formato mp4. O ficheiro *manifest* é um ficheiro em XML que contém informação acerca da informação desses

segmentos, incluindo, *bit rate*, duração, instâncias de vídeo e áudio. Com esta informação, o cliente sabe, quais os segmentos a pedir, qual o vídeo a invocar se for necessário diferente *bit rate* devido a mudança da *bandwidth* da rede. É possível observar um exemplo de um ficheiro *manifest* em MOG Manifest File.

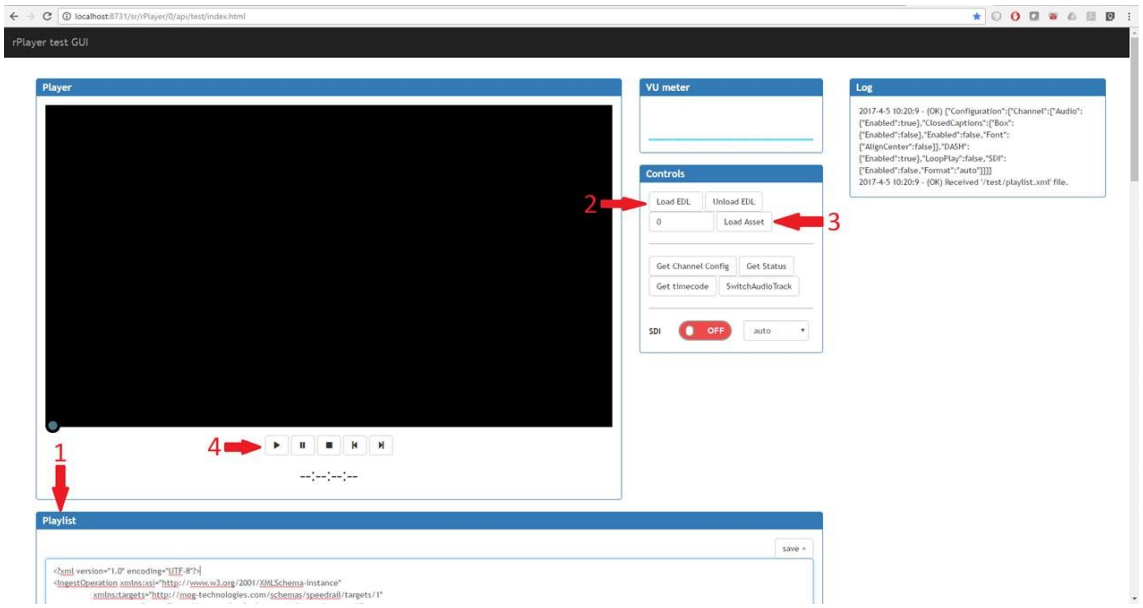

Figura 5.9: GUI RPlayer

Capítulo 6

Resultados e Discussão

Nesta capítulo é apresentado uma simples interface web e, de seguida, o ambiente de teste, as métricas utilizada, os resultados e a discussão dos mesmo.

6.1 Interface Web

Foi desenvolvida uma simples interface *web*, em *python* de maneira a ser possível reproduzir os segmentos do vídeo final. Para esse efeito, foi utilizado o leitor Dash.JS [1] , uma biblioteca *open source* em *JavaScript* que permite reproduzir DASH.

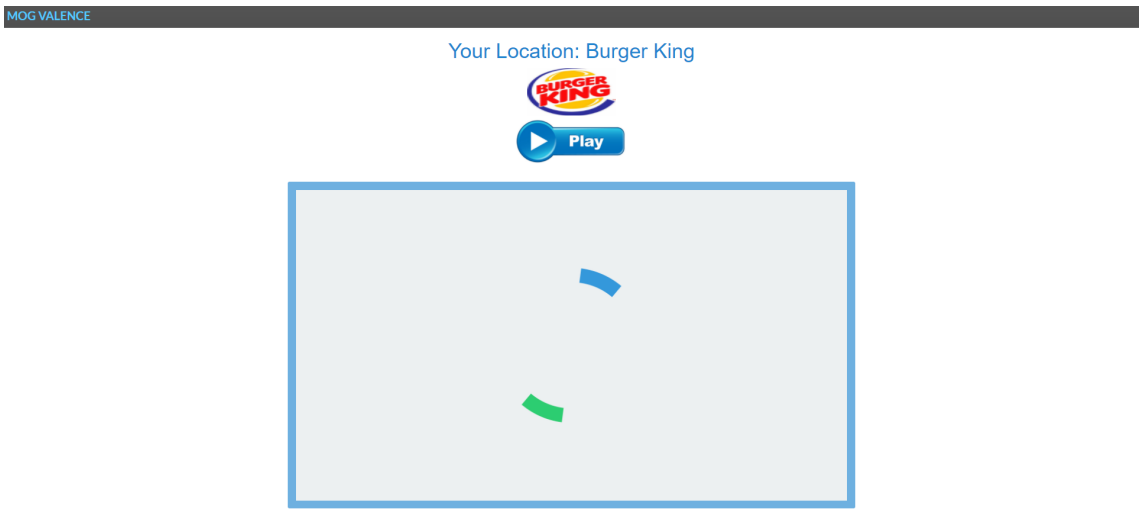

Figura 6.1: Interface *Web*

A interface desenvolvida 6.1 menciona a localização do visualizador e permite reproduzir o vídeo final de acordo com a localização do visualizador. Esta interface foi implementada em

[1] https://github.com/Dash-Industry-Forum/dash.js

HTML, com o uso de CSS para aspetos visuais. A utilização de JavaScript também foi necessária de modo a ser possível reproduzir o vídeo, efetuando pedidos HTTP. A localização do visualizador é distinguida e obtida através do IP de quem fez o pedido para visualizar o vídeo, o cliente.

6.2 Testes e Validação

De maneira a testar a implementação, definiu-se a simulação de um caso de uso real, sendo, desta maneira possível validar a aplicabilidade do protótipo e dos módulos desenvolvidos num ambiente a que estaria sujeito na realidade.

Passamos o mesmo vídeo para diferentes utilizadores, com localizações diferentes e teremos que observar se o anúncio é diferente para cada um deles.

Podemos usar o seguinte caso de uso 6.2, onde o mesmo vídeo será passado em duas grandes empresas, como o Mc Donalds e o Burger King. Porém, no Mc Donalds o vídeo passará com publicidade da Coca-Cola enquanto que o mesmo vídeo no Burger King passará anúncios da Pepsi. Para esta simulação, foi criada uma rede *Wireless Ad Hoc*. Isto possibilita a partilha da informação armazenada num computador com outro dispositivo e, assim ser possível, ter dois dispositivos com endereços diferentes a aceder à aplicação, e desse jeito, distinguir qual a publicidade a mostrar no vídeo.

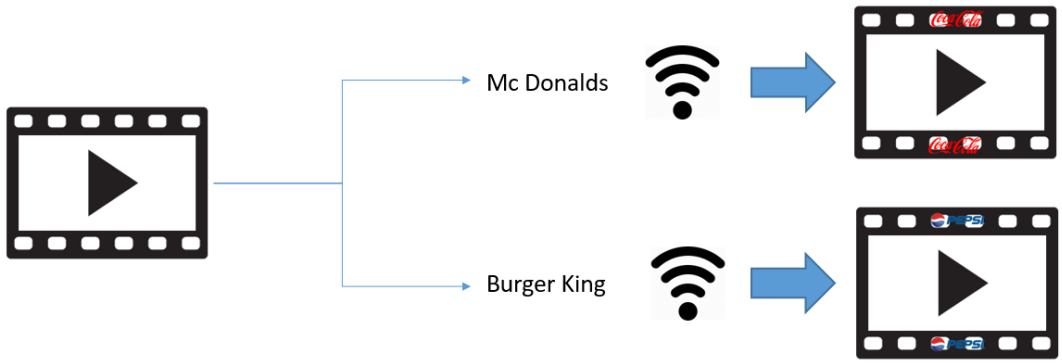

Figura 6.2: Caso de uso

6.2.1 Ambiente de Teste

Na tabela 6.1 estão descritas as principais características da máquina de testes.

Tabela 6.1: Características da máquina de teste

Característica	PC
CPU	Intel Core I7-6700 @ 3.40 GHz
Memória RAM	8 GB
Sistema Operativo	Windows 10 Enterprise
Placa de Rede	Intel Ethernet I219-LM

Foram realizados testes com quatro vídeos diferentes, cujas principais características estão descritas na tabela 6.2 .

Tabela 6.2: Características dos vídeos de teste

Característica	Vídeo 1	Vídeo 2	Vídeo 2.1	Vídeo 3
Tamanho em Disco	3.17 GB	134 MB	134 MB	186 MB
Duração	8m14s	18s	18s	25 s
Dimensões	1920x1080	1920x1080	1920x1080	720x576
Codec Comercial	XDCAM HD422	XDCAM HD422	XDCAM HD422	IMX 50
Bit rate	50 MB/s	50 MB/s	50 MB/s	50 MB/s
Números de Placements	1	1	3	1

Foram utilizadas as seguintes métricas de maneira a analisar performance do protótipo desenvolvido. Estes são testes em relação à primeira parte da arquitetura.

- Utilização da memória RAM.
- Utilização do CPU.
- Tempo de construção do vídeo.

6.2.2 Utilização da memória RAM e do CPU

Nos seguintes diagramas 6.3 e 6.4 é possível observar a utilização da memória RAM e do CPU da aplicação quando está a ser construído o vídeo final, ou seja, o vídeo original incorporado com os anúncios NIVA.

Podemos concluir que este processo requer uma baixa necessidade de memória RAM em qualquer um dos processos dos vídeos, sendo que a dimensão do vídeo e o número de anúncios a inserir tem uma influencia direta na utilização da memória RAM. Logo quanto melhor for a dimensão do vídeo e quanto maior for o número de inserções de anúncios mais RAM será utilizada.

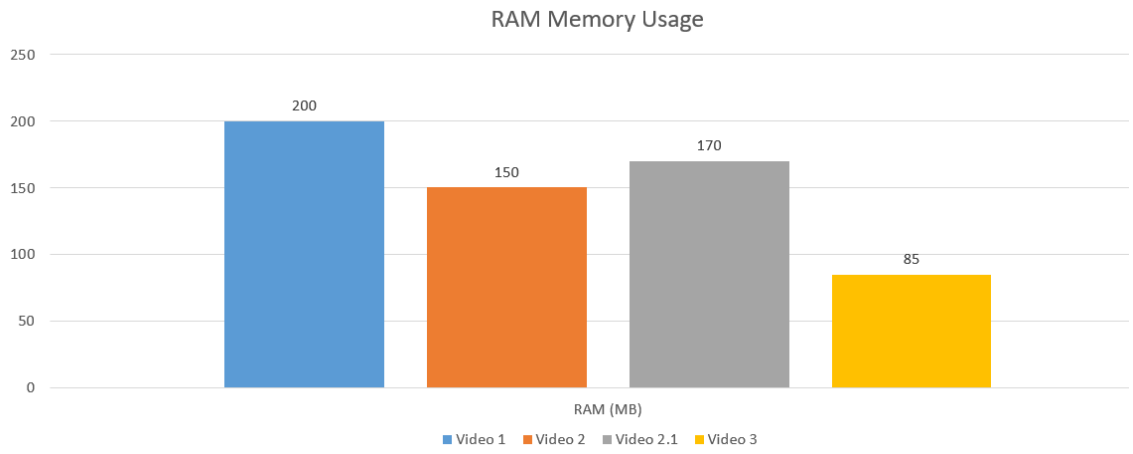

Figura 6.3: Uso da RAM

Tendo em conta os resultados obtidos, a utilização do CPU é uma componente crítica, esta poderá precisar de melhoramentos de forma a reduzir a quantidade dos recursos necessários. De notar que neste caso, também, podemos afirmar que as dimensões do vídeos estão relacionados com o consumo de CPU contudo o número de inserções de anúncios, praticamente, não influencia.

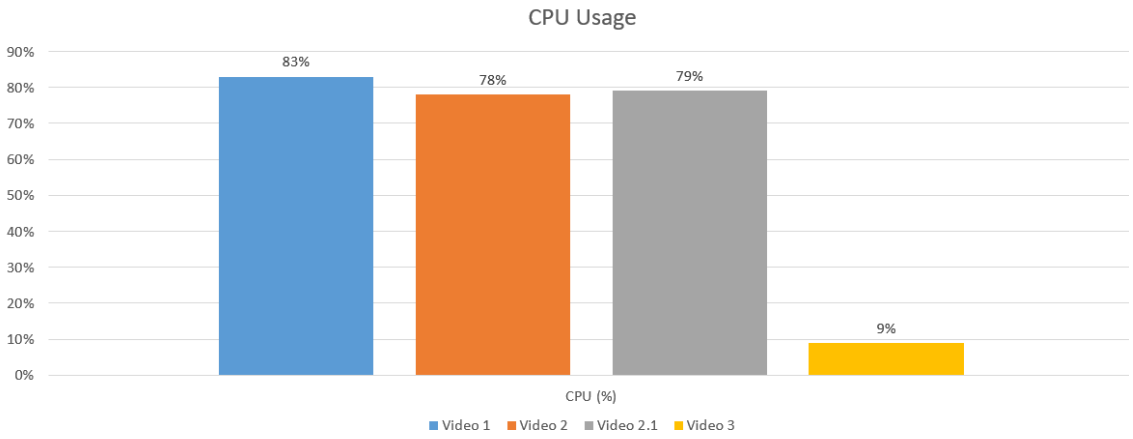

Figura 6.4: Uso do CPU

6.2.3 Tempo para construir vídeo

No diagrama seguinte 6.5, é possível verificar o tempo que demora a construir o vídeo final com anúncios. Este tempo é devido, principalmente, ao processo de codificação e descodificação, que está diretamente relacionado com a qualidade do vídeo, duração do vídeo e do *codec* utilizado. Também podemos afirmar que quanto maior for o número de inserções, maior será o tempo.

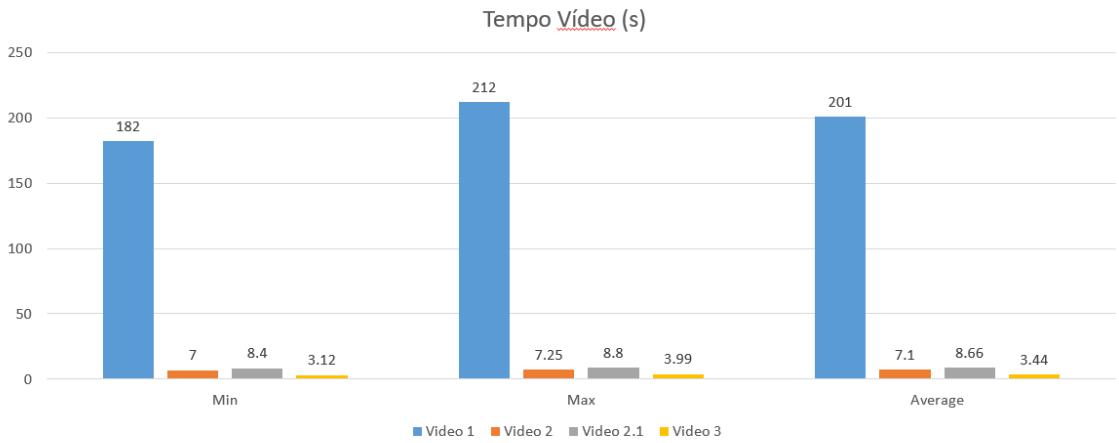

Figura 6.5: Tempo de Construção do Vídeo (s)

Capítulo 7

Conclusões e Trabalho Futuro

Esta dissertação descreve uma arquitetura de alto nível para fazer *product placement* em televisão, contextualizando os anúncios baseados na localização do utilizador. Verificou-se, através do protótipo desenvolvido, que esta arquitetura poderá, facilmente, ser expandida, tornando-a numa solução possível de ser trabalhada e melhorada futuramente.

As tradicionais paragens para os anúncios na televisão estão a morrer. A nova forma de publicidade é designada por *Native in Video Advertising*, onde os objetos a serem promovidos aparecem dentro do conteúdo do programa. Esta solução combina *Native in Video Advertising*, publicidade personalizada e *Location Based Services*.

Quando da visualização do vídeo, todos os telespectador irão ver o mesmo vídeo a maior parte do tempo mas os segmentos com os anúncios serão servidos de uma forma diferente dependendo da localização do mesmo.

7.1 Trabalho Realizado e Satisfação dos Objetivos

Os objetivos inicialmente propostos foram atingidos com sucesso. O trabalho realizado permitiu desenvolver um novo produto, algo que não existia até ao momento, e bastante útil para a empresa, *MOG Technologies*.

A arquitetura desenvolvida permite que, facilmente, o trabalho realizado possa ser estendido no futuro, possibilitando, por exemplo, desenvolver novos módulos. Sendo que, os módulos desenvolvidos, foram construídos de forma a permitir a interoperabilidade entre si e entre qualquer produto de software da *MOG Technologies*. O protótipo desenvolvido permitiu, com sucesso, testar a arquitetura proposta.

7.2 Trabalho Futuro

Depois de já se ter feito algumas prova de conceitos da implementação desenvolvida, é necessário ainda, algum trabalho de desenvolvimento para que se possa lançar um produto no mercado,

com as exigências que este requer. Estando esta aplicação a correr em ambiente *cloud*, irá permitir uma investigação sobre mecanismos que permitam o aumento da eficiência do processamento.

Mesmo não sendo o objetivo desta dissertação, seria interessante analisar a velocidade das operações de codificação e descodificação no *Demuxer*, pois são as operações mais críticas no sistema. Assim, é bastante conveniente estudar, por um lado, otimizações possíveis a esses processos, e por outro, diferenças de performance para diferentes *codecs*.

Outro problema detetado deve-se ao facto de os leitores de MPEG-DASH atuais serem bastante limitados ao nível dos formatos que suportam, ou seja, cada um teu o seu ficheiro MPD. Seria bastante interessante realizar trabalhos no sentido de alargar os formatos por eles suportados.

Neste momento, o processo da construção do vídeo não é um processo em tempo real. A questão mais crítica a considerar deste protótipo é o elevado armazenamento necessário, isto porque, os vídeos precisam de ser pré-processados e armazenadas antes de estarem disponíveis para os utilizadores. No entanto, se este processo fosse feito todo em tempo real, seria necessário um elevado poder de processamento, para se conseguir dar uma rápida resposta ao visualizador no tempo pretendido.

Anexo A

Anexos

Depois das conclusões e antes das referências bibliográficas, apresenta-se neste anexo o artigo cientifico e um conjunto de imagens de maneira a suportar a dissertação.

A.1 Paper

New Cloud Services for product placement in television*

†

André Regado
MOG Technologies
Porto, Portugal
andre.regado@mog-technologies.com

Alexandre Ulisses
MOG Technologies
Porto, Portugal
alexandre.ulisses@mog-technologies.com

Miguel Poeira
MOG Technologies
Porto, Portugal
miguel.poeira@mog-technologies.com

Pedro Santos
MOG Technologies
Porto, Portugal
pedro.santos@mog-technologies.com

ABSTRACT

Over the last years, video consumption under digital format is increasing. There are more people watching television through their tablet's, smartphones and computers. This allows that adverts maybe targeted according the final user preferences. The goal of this work is to present a cloud based platform that allows an insertion of personalized and contextualized ads, in real time, based on the end user network location.

CCS CONCEPTS

•**Information systems → Multimedia streaming;** *Computational advertising;*

KEYWORDS

Native Advertising, Advertising, Location Based Services, Cloud Computing, Adaptive Bitrate Streaming

ACM Reference format:
André Regado, Alexandre Ulisses, Miguel Poeira, and Pedro Santos. 2017. New Cloud Services for product placement in television. In *Proceedings of ACM International Conference On Interactive Experiences for Television and Online Video, Hilversumlands, March 2017 (TVX'17),* 4 pages.
DOI: 10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

The increase of video consumption under digital format is evident. Nowadays everyone has a smartphone or tablet, where they can watch television. Usually, these devices have Internet access which has lead to a substantial growth of the online video marketing market in the last few years.

Native Advertising is the new way to display advertising, where the ads will be embedded in the video content so that the viewer can't reject the advertisements' messages.

The registration of the user in a specific wifi network, for watching videos, defines a context that allows personalized adverts streaming.

In this paper, we present a high-level architecture for contextualize product placement in online television using a cloud-based platform.

2 ADVERTISING IN VIDEO

The way to do advertising in video is changing. Traditionally, adverts appeared on Television in ad breaks between TV program segments, where the use of pre-roll, mid-roll and post-roll are well known. However, the traditional ad breaks became ineffective and inefficient, so it's crucial to change the way Tv ads are displayed. Native Advertising is a concept whose main goal is to improve the way we display ads, putting an end to the concept of advertisement breaks, by embedding them in the content of the video itself [6, 10].

We can apply Native Advertisement in video. However, this poses a complex problem as each part of the program has an unique way to display its content. A possible way to do this is by replacing the objects embedded in the video according the final user.The incorporation of these objects in the video must be done in a seamless way in order to avoid any kind of interference with the main content. To achieve this naturalness, computer vision techniques are needed to match the object into the content namely by controlling the texture and lightning of the new object.

3 PERSONALISED ADVERTISEMENTS IN VIDEO

In this work, two approaches for personalized advertising were combined: Contextual Match Advertising and Location Based Advertising [13]. The proposed architecture will fuse different objects into the video content that, somehow, will be related with it. The choice of objects to be embedded will be based on Location-based services (LBS), which allow us to know where the end user device is [7, 11].

The usage of location-based advertising is possible due to the fact that any place where the user goes, he will usually carry a smart device with him. Location-Based Advertising is efficient because, first of all is contextualized as ads could be personalized according

*Produces the permission block, and copyright information
†The full version of the author's guide is available as `acmart.pdf` document

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
TVX'17, Hilversumlands
© 2017 ACM. 978-x-xxxx-xxxx-x/YY/MM…$15.00
DOI: 10.1145/nnnnnnn.nnnnnnn

Figura A.1

to user location. Besides that, it is timely because location data is recorded in real time and it allows brands to reach people on a specific precise moment [5]. Nowadays, location based services are already used in various applications [3, 14].

- Marketing: companies use user's location to send messages to them.
- Information Services: when a user wants to know the closest restaurant or cinema.
- Navigation: navigation services allows the user to pinpoint its exact geographical coordinates and get the directions to the required location.
- Hobby: there are games that take advantage of the user's location for gamification purposes (e.g. Pokemón GO).

4 HIGH-LEVEL ARCHITECTURE

In this section, we provide a high-level architecture for product placement in television using a cloud-based platform. We will present how the system works and how its components interact.

4.1 MPEG-DASH

MPEG-DASH is an Adaptive Bitrate Streaming technique, it works by breaking the content into a sequence of small HTTP-based file segments. The server will provide the video content in multiple different bit rates. In the beginning of the session, a client requests a Media Presentation Description (MPD) to the Server. While the DASH client is buffering and playing the video content, it will also analyse the variation of the network bandwidth and, depending on this analysis, it will decide which segments to download to maintain the appropriate buffering.

In summary, DASH utilizes an algorithm that uses the maximum streaming bit rate that the network allows in order to deliver the content with the expected Quality of Experience (QoE) to the user [15, 16].

By taking advantage of cloud's benefits, we are proposing an architecture that is [1, 2, 9]:

- Built on modular components and blocks, that can be re-utilized to build new functionalities.
- Expansible, which means that it can be adapted to new scenarios.
- Scalable, that is, operation's needs are not bound by hardware stock availability or ease of deployment.
- Collaborative, because it enables people to easily collaborate on the same project, wherever they are located.
- Enables fast, transparent updates and bug fixes, which lets the production team abstract on technical details.

4.2 Metadata

Video Multiple Ad Playlist (VMAP) is used to express the structure of the ad inventory as a set of timed ad breaks within a publisher's video content [4].

In the following image 2, we can see an example of one AdBreak in VMAP. The most important information is the time offset because it indicates the exact time that this segment needs to be inserted in the video content. One other important fact is that combined with the VMAP structure we receive a VAST response which has more information about the advertisement, such as its duration and location.

```
<vmap:AdBreak breakType="linear" breakId="myid" timeOffset="00:00:04">
    <vmap:AdSource allowMultipleAds="false" followRedirects="true" id="1">
        <vmap:VASTAdData>
            ...
        </vmap:VASTAdData>
    </vmap:AdSource>
    <vmap:TrackingEvents>
        <vmap:Tracking event="breakStart">
            http://MyServer.com/breakstart.gif
        </vmap:Tracking>
    </vmap:TrackingEvents>
</vmap:AdBreak>
```

Figure 2: VMAP example

Video Ad Serving Template (VAST) is a video ad-serving template that provides a uniform way for advertising data to be transferred from ad servers to video players independent of any technology. In others words, VAST provides a common protocol that enables ad servers to use a single ad format across multiple video players [8].

In the following image 3, we can see an example of VAST. Through VAST we have information about the location and the duration of the ad:

```
<Ad>
    <InLine>
        <AdSystem>
            My Ad Server
        </AdSystem>
        <AdTitle>
            Car Company
        </AdTitle>
        <Description>
            Brand
        </Description>
        <Impression id="myid">
            ...
        </Impression>
        <Creatives>
            <Creative>
                <Linear>
                    <Duration>
                        00:05:23.125
                    </Duration>
                    <TrackingEvents>
                        <Tracking event="start">
                            http://MyServer.com/breakstart.gif
                        </Tracking>
                    </TrackingEvents>
                    <MediaFiles>
                        ...
                    </MediaFiles>
                </Linear>
            </Creative>
        </Creatives>
    </InLine>
<Ad>
```

Figure 3: VAST example

4.2.1 XML Schema (XSD) will be used to express a set of rules to which VMAP and VAST must obey in order to be considered valid according to that schema.

Figura A.2

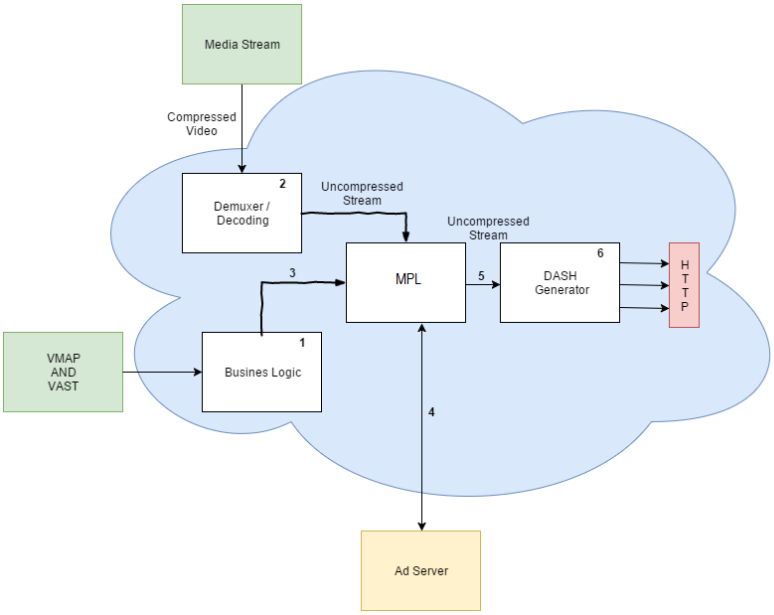

Figure 1: Architecture. Components and its interactions

4.3 Components

In this section, we present four components that compose our architecture 1. it will be demonstrated how the architecture will switch the original video with the ads according to the final user profile.

Native Advertisement is too computational intensive to be delivered live, so we see the initial systems having multiple parallel sequences of a program, each with different objects/items placed in them. So it is not a real time process because, for realism, it is necessary to consider shadows, reflections, among others, which need to be pre-processed.

The Ad Server will store the program segments in multiple versions, where the same short piece of program material has a different product placed in each version. This server needs to be connection wise, close to the program servers or CDN's. The goal of a CDN is to provide content to end-users focusing on improving its availability and performance by forming a system of cache servers that use geographical proximity as a criteria of how to deliver that content [12].

4.3.1 Business Logic receives as an input two XML's (VAST and VMAP). It's responsible for parsing both XML's and for validating these XML's against their XSD (XML Schema). Then, Business Logic transfers the VMAP and VAST information to MPL..

4.3.2 Demuxer receives as an input the compressed original video and deliveries as its output the uncompressed video.

4.3.3 MPL receives as an input the uncompressed stream from the demuxer module and the information of VMAP and VAST from the Business Logic module. The main goal of this component is to switch between the main program and the correct segments.

4.3.4 Dash Generator receives as an input an uncompressed stream and as the name depicts, it is responsible for generating different representations (segments) and providing them.

4.4 WorkFlow

In this section, we provide more details about the components that compose the proposed architecture 1 and how they interact each other.

(1) Business Logic processes VMAP and VAST. To do this, it was used the Microsoft XML Core Services (MSXML), a set of services that allows xml-based applications. MSXML uses Document Object Model (DOM), to handle a XML document as a tree structure where each node is an object representing a part of the document. XPath was used to identify and navigate through nodes by using path syntax. It was also used to validate XML documents using XSD.

Figura A.3

(2) Demuxer receives the original stream. A demuxer is a module responsible for splitting individual elements of a media file, e.g., video, audio or subtitles and sending them to their respective decoders. To do this, FFmpeg was used as an open source project for handling multimedia data like encoding and decoding.
(3) Through VMAP, Business Logic informs the MPL about the number of segments that need to be changed, the exact time that it is needed to insert a segment embedded with ads and its duration and location by VAST.
(4) MPL is the brain of the system. It is responsible for receiving information from other nodes and for switching streams between the original video and the segments with the most appropriate item placed in it and getting back to the original program stream. MPL downloads the replacements segments from Ad Server. Our approach is to have a library of many short sections of the program, each with a different product placement in it. These will be stored in an Ad Server and the user's location identity will cause the recall of the most contextually appropriate sequence of placement.
(5) From Business Logic information and the original video from demuxer module, Switch outputs the correct video with the ads.
(6) DASH Generator breaks the original video content into HTTP-based segments/chunks (2 seconds) and encodes it at multiple bit rate, streaming them to the associated CDN.

5 CONCLUSION

This paper has described a high-level architecture for product placement in television using a cloud-based infrastructure. Traditional advertising breaks are dying. The new form of advertising is Native Advertising, where the item to be promoted appears placed within the program. This solution combines Native Advertising, personalized advertisements and location based services.

The viewer will see the program in real time, but the architecture will be switching streams seamlessly between main program scenes, selected stream of program with most appropriate item placed in it and back to the main program stream. When watching a video, for example, all viewers will see the same video most of the time, but the placement scenes will be served differently depending on the viewer's location.

REFERENCES

[1] Majed Almubaddel and Ahmed M Elmogy. 2016. Cloud Computing Antecedents, Challenges, and Directions. *Proceedings of the International Conference on Internet of Things and Cloud Computing* (2016), 16:1–16:5. DOI: http://dx.doi.org/10.1145/2896387.2896401

[2] Ahmed Alzahrani, Nasser Alalwan, and Mohamed Sarrab. 2014. Mobile cloud computing: advantage, disadvantage and open challenge. *Proceedings of the 7th Euro American Conference on Telematics and Information Systems - EATIS '14* (2014), 1–4. DOI: http://dx.doi.org/10.1145/2590651.2590670

[3] Aleksander Buczkowski. 2012. Location-Based Services - Applications. (2012). http://geoawesomeness.com/knowledge-base/location-based-services/location-based-services-applications/

[4] Interactive Advertising Bureau. 2014. Video Multiple Ad Playlist (VMAP). (2014). http://www.iab.com/wp-content/uploads/2015/06/VMAP.pdf

[5] Hung Dang and Ee-Chien Chang. 2015. PrAd: Enabling Privacy-Aware Location Based Advertising. *Proceedings of the 2Nd Workshop on Privacy in Geographic Information Collection and Analysis* c (2015), 1:1–1:8. DOI: http://dx.doi.org/10.1145/2830834.2830839

[6] IAB. 2016. Digital Video In-Stream Ad Format Guidelines and Best Practices [Report]. May (2016), 4–23. http://www.iab.com/wp-content/uploads/2016/01/DVAFG

[7] Amit Kushwaha and Vineet Kushwaha. 2011. Location Based Services using Android mobile Operating System. *International Journal of Advances in Engineering & Technology* 1, 1 (2011), 14–20. http://www.ijaet.org/media/Location-Based-Services-using-Android-mobile-Operating-System-Copyright-IJAET.pdf

[8] R E V W Illiam R C Laytor, Betsy Currie, Rhonda Stubbs, Nancy Smith, Jenna Robinson, A M Jerry Broome, Mark Mitchell, Steve Thrailkill, and Ken Troutman. 2012. Video Ad Serving Template (VAST). (2012), 71. https://www.iab.com/wp-content/uploads/2015/06/VASTv3

[9] Peter Mell and Timothy Grance. 2011. The NIST definition of cloud computing. *NIST Special Publication* 145 (2011), 7. DOI: http://dx.doi.org/10.1136/emj.2010.096966

[10] Joe Pulizzi. 2014. The Ultimate Guide to Native Advertising. (2014). https://www.linkedin.com/pulse/20140107180859-5853751-the-ultimate-guide-to-native-advertising

[11] Ana-Maria Roxin, Christophe Dumez, Maxime Wack, and Jafaar Gaber. 2008. Middleware models for location-based services. *Proceedings of the 2nd international workshop on Agent-oriented software engineering challenges for ubiquitous and pervasive computing - AUPC '08* (2008), 35–40. DOI: http://dx.doi.org/10.1145/1387249.1387255

[12] M Zubair Shafiq, Alex X Liu, and Amir R Khakpour. 2012. Revisiting Caching in Content Delivery Networks. *Sigmetrics* (2012), 567–568. DOI: http://dx.doi.org/10.1145/2637364.2592021

[13] Maad Shatnawi and Nader Mohamed. 2012. Statistical techniques for online personalized advertising. *Proceedings of the 27th Annual ACM Symposium on Applied Computing* (2012), 680–687. DOI: http://dx.doi.org/10.1145/2245276.2245406

[14] Jungwon Min Shu Wang and Byung K Yi. 2008. Location Based Services for Mobiles: Techonogies and Stardards. January 2008 (2008). https://www.researchgate.net/publication/265758564

[15] Viswanathan Swaminathan. 2013. Are we in the middle of a video streaming revolution? *ACM Transactions on Multimedia Computing, Communications, and Applications* 9, 1s (2013), 1–6. DOI: http://dx.doi.org/10.1145/2490826

[16] Christian Timmerer and C Griwodz. 2012. Dynamic adaptive streaming over HTTP: from content creation to consumption. *Proceedings of the 20th ACM international …* (2012), 1533–1534. DOI: http://dx.doi.org/10.1145/2393347.2396553

Figura A.4

A.2 MOG_COD

```
<IngestOperation xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:
    targets="http://mog-technologies.com/schemas/speedrail/targets/1" xmlns:st="
    http://mog-technologies.com/schemas/storage/1" xmlns:me="http://mog-
    technologies.com/schemas/metadata/1" xmlns:mea="http://mog-technologies.com/
    schemas/metadata/avid/1" xmlns:mam="http://mog-technologies.com/schemas/
    asset_management/1" xmlns:media="http://mog-technologies.com/schemas/data_types
    /media/1" xmlns:avs="http://mog-technologies.com/schemas/storage/avid/1" xmlns:
    ama="http://mog-technologies.com/schemas/asset_management/avid/1" xmlns:auth="
    http://mog-technologies.com/schemas/authentication/1" xmlns:options="http://mog
    -technologies.com/schemas/speedrail/options/1" xmlns:qc="http://mog-
    technologies.com/schemas/speedrail/quality_control/1" xmlns="http://mog-
    technologies.com/schemas/speedrail/operations/ingest_operation/1">

 <Channels>
  <Channel>
   <Targets>
    <targets:OP1a name="OportonityCity">
     <options:AudioOptions>
      <options:ChannelMapping>
       <options:Mode value="sameAsSource" />
       <options:Channels>
        <options:OutputChannel inputChannel="1" pcmAudioDescriptor="
            sameAsSource" />
        <options:OutputChannel inputChannel="2" pcmAudioDescriptor="
            sameAsSource" />
        <options:OutputChannel inputChannel="3" pcmAudioDescriptor="
            sameAsSource" />
        <options:OutputChannel inputChannel="4" pcmAudioDescriptor="
            sameAsSource" />
        <options:OutputChannel inputChannel="5" pcmAudioDescriptor="
            sameAsSource" />
        <options:OutputChannel inputChannel="6" pcmAudioDescriptor="
            sameAsSource" />
        <options:OutputChannel inputChannel="7" pcmAudioDescriptor="
            sameAsSource" />
        <options:OutputChannel inputChannel="8" pcmAudioDescriptor="
            sameAsSource" />
        <options:OutputChannel inputChannel="9" pcmAudioDescriptor="
            sameAsSource" />
        <options:OutputChannel inputChannel="10" pcmAudioDescriptor="
            sameAsSource" />
        <options:OutputChannel inputChannel="11" pcmAudioDescriptor="
            sameAsSource" />
        <options:OutputChannel inputChannel="12" pcmAudioDescriptor="
            sameAsSource" />
        <options:OutputChannel inputChannel="13" pcmAudioDescriptor="
            sameAsSource" />
        <options:OutputChannel inputChannel="14" pcmAudioDescriptor="
            sameAsSource" />
```

```
25            <options:OutputChannel inputChannel="15" pcmAudioDescriptor="
                  sameAsSource" />
26            <options:OutputChannel inputChannel="16" pcmAudioDescriptor="
                  sameAsSource" />
27          </options:Channels>
28          <options:ChannelMissingBehaviour value="exclude" />
29          <options:Aligned value="false" />
30        </options:ChannelMapping>
31      </options:AudioOptions>
32      <targets:AllowOverwrite value="true" />
33      <targets:Storage>
34        <st:SmbCifsDirectory>
35          <st:serverAddress>PC-aregado</st:serverAddress>
36          <auth:Auths>
37            <auth:UsernamePassword>
38              <auth:user>mogro@mog</auth:user>
39              <auth:password>m0gr0</auth:password>
40            </auth:UsernamePassword>
41          </auth:Auths>
42          <st:fullPath>\Shared\sources\edl</st:fullPath>
43        </st:SmbCifsDirectory>
44      </targets:Storage>
45      <targets:CreateEmptyClosedCaption value="false" />
46    </targets:OP1a>
47  </Targets>
48 </Channel>
49 </Channels>
50 <Logs path="C:\ProgramData\MOG\mxfSPEEDRAIL\Logs Operations" level="info" />
51<Edls>
52  <Merge kind="asFile" />
53  <IndexedSequenceClips value="false" />
54  <Edl>
55    <DiscardDataTracks value="false" />
56    <PreserveOriginalTimecode value="false" />
57    <Track kind="compound">
58      <Clip kind="smpte_mxf">
59        <media:CutPoints>
60          <media:RelativeTimecodeCutPoints in_point="00:00:00:00" out_point="
                00:00:08:00" />
61        </media:CutPoints>
62        <Source>
63          <st:SmbCifsFile>
64            <st:serverAddress>pc-aregado</st:serverAddress>
65            <st:fullPath>\d\__TESTS__\AR\OportonityCity.MXF</st:fullPath>
66          </st:SmbCifsFile>
67        </Source>
68      </Clip>
69      <Clip kind="smpte_mxf">
70        <media:CutPoints>
```

```
71          <media:RelativeTimecodeCutPoints in_point="00:00:00:00" out_point="
                00:00:03:00" />
72        </media:CutPoints>
73        <Source>
74          <st:SmbCifsFile>
75            <st:serverAddress>pc-aregado</st:serverAddress>
76            <st:fullPath>\d\__TESTS__\AR\AdMacDonals.MXF</st:fullPath>
77          </st:SmbCifsFile>
78        </Source>
79      </Clip>
80      <Clip kind="smpte_mxf">
81        <media:CutPoints>
82          <media:RelativeTimecodeCutPoints in_point="00:00:11:00" out_point="
                00:00:15:00" />
83        </media:CutPoints>
84        <Source>
85          <st:SmbCifsFile>
86            <st:serverAddress>pc-aregado</st:serverAddress>
87            <st:fullPath>\d\__TESTS__\AR\OportonityCity.MXF</st:fullPath>
88          </st:SmbCifsFile>
89        </Source>
90      </Clip>
91    </Track>
92    <UmidOptions>
93      <SourceUmid>
94        <Automatic usage="preserveFromSource" />
95      </SourceUmid>
96      <TargetUmid>
97        <Automatic usage="generateNew" />
98      </TargetUmid>
99    </UmidOptions>
100   <Metadata>
101     <me:Flat />
102     <mea:Locators />
103   </Metadata>
104  </Edl>
105 </Edls>
106 </IngestOperation>
```

A.3 VMAP

```
1  <vmap:AdBreak breakType="linear" breakId="mypre" timeOffset="00:00:06:00">
2    <vmap:AdSource allowMultipleAds="false" followRedirects="true" id="1">
3      <vmap:CustomAdData templateType="vast1">
4              <![CDATA[preroll.xml]]>
5          </vmap:CustomAdData>
```

```
6     </vmap:AdSource>
7     <vmap:TrackingEvents>
8       <vmap:Tracking event="breakStart">
9         http://MyServer.com/breakstart.gif
10      </vmap:Tracking>
11    </vmap:TrackingEvents>
12  </vmap:AdBreak>
```

A.4 VAST

```
1  <Ad>
2     <InLine>
3       <AdSystem>
4         My Ad Server
5       </AdSystem>
6       <AdTitle>
7         Car Company
8       </AdTitle>
9        <Description>
10        Brand
11      </Description>
12      <Impression id="asd1">
13              <![CDATA[preroll.xml]]>
14      </Impression>
15      <Creatives>
16        <Creative>
17          <Linear>
18            <Duration>
19              00:00:02:00
20            </Duration>
21            <TrackingEvents>
22              <Tracking event="start">
23                http://MyServer.com/breakstart.gif
24              </Tracking>
25            </TrackingEvents>
26            <MediaFiles>
27              <MediaFile id="zxc" delivery="streaming" type="MIME" bitrate="750"
                    minBitrate="400" maxBitrate="1000" width="720" height="640" scalable
                    ="true" maintainAspectRatio="true" codec="H.264">
28                <![CDATA[preroll.xml]]>
29              </MediaFile>
30              <MediaFile id="zwac" delivery="progressive" type="MIME" bitrate="860"
                    minBitrate="720" maxBitrate="1200" width="1024" height="860"
                    scalable="true" maintainAspectRatio="false" codec="H.264">
31                <![CDATA[midroll.xml]]>
32              </MediaFile>
```

```
33          </MediaFiles>
34        </Linear>
35      </Creative>
36    </Creatives>
37   </InLine>
38  </Ad>
```

A.5 MOG Manifest File

```
1  <Period start="PT0.0S">
2    <AdaptationSet contentType="video" segmentAlignment="true" bitstreamSwitching="
         true" frameRate="25/1">
3      <Representation id="0" mimeType="video/mp4" codecs="avc1.64002a" bandwidth="
           5000000" width="1280" height="720" frameRate="25/1">
4        <SegmentTemplate timescale="1000000" duration="520000" initialization="init-
             stream$RepresentationID$.m4s" media="chunk-stream$RepresentationID$-
             $Number%08d$.m4s" startNumber="1">
5        </SegmentTemplate>
6      </Representation>
7    </AdaptationSet>
8    <AdaptationSet contentType="audio" segmentAlignment="true" bitstreamSwitching="
         true">
9      <Representation id="1" mimeType="audio/mp4" codecs="mp4a.40.2" bandwidth="
           96000" audioSamplingRate="48000">
10       <AudioChannelConfiguration schemeIdUri="urn:mpeg:dash:23003:3:
             audio_channel_configuration:2011" value="2" />
11       <SegmentTemplate timescale="1000000" duration="520000" initialization="init-
             stream$RepresentationID$.m4s" media="chunk-stream$RepresentationID$-
             $Number%08d$.m4s" startNumber="1">
12       </SegmentTemplate>
13     </Representation>
14     <Representation id="2" mimeType="audio/mp4" codecs="mp4a.40.2" bandwidth="
           96000" audioSamplingRate="48000">
15       <AudioChannelConfiguration schemeIdUri="urn:mpeg:dash:23003:3:
             audio_channel_configuration:2011" value="2" />
16       <SegmentTemplate timescale="1000000" duration="520000" initialization="init-
             stream$RepresentationID$.m4s" media="chunk-stream$RepresentationID$-
             $Number%08d$.m4s" startNumber="1">
17       </SegmentTemplate>
18     </Representation>
19   </AdaptationSet>
20  </Period>
```

Referências

[AL12] Jens Abrahamsson e Niclas Lindblom. *Product Placement*. PhD thesis, 2012.

[And97] David P. Anderson. Device reservation in audio/video editing systems. *ACM Transactions on Computer Systems*, 15(2):111–133, 1997. URL: http://dl.acm.org/citation.cfm?id=253145.253149, doi:10.1145/253145.253149.

[AU14] B M Ashwini e J Usha. Location Based Services - Positioning Techniques and its Applications. 3(1):176–183, 2014. URL: http://www.ijaiem.org/volume3issue1/IJAIEM-2014-01-20-042.pdf.

[BNVdB04] Geert Jan Bex, Frank Neven e Jan Van den Bussche. DTDs versus XML Schema: A Practical Study. *Proceedings of the 7th International Workshop on the Web and Databases colocated with ACM SIGMOD/PODS 2004 - WebDB '04*, (WebDB):79–84, 2004. URL: http://portal.acm.org/citation.cfm?doid=1017074.1017095, doi:10.1145/1017074.1017095.

[Buc12] Aleksander Buczkowski. Location-Based Services - Applications, 2012. URL: http://geoawesomeness.com/knowledge-base/location-based-services/location-based-services-applications/.

[Bur14] Interactive Advertising Bureau. Video Multiple Ad Playlist (VMAP). 2014. URL: http://www.iab.com/wp-content/uploads/2015/06/VMAP.pdf.

[Bus] Business Insider. Native ads will drive 74% of all ad revenue by 2021. URL: http://www.businessinsider.com/the-native-ad-report-forecasts-2016-5.

[CLWH16] Zhi-Qi Cheng, Yang Liu, Xiao Wu e Xian-Sheng Hua. Video eCommerce: Towards Online Video Advertising. *Proceedings of the 2016 ACM on Multimedia Conference*, pages 1365–1374, 2016. URL: http://doi.acm.org/10.1145/2964284.2964326, doi:10.1145/2964284.2964326.

[DC15] Hung Dang e Ee-Chien Chang. PrAd: Enabling Privacy-Aware Location Based Advertising. *Proceedings of the 2Nd Workshop on Privacy in Geographic Information Collection and Analysis*, (c):1:1–1:8, 2015. URL: http://doi.acm.org/10.1145/2830834.2830839, doi:10.1145/2830834.2830839.

[FMB12] Jeremy D. Foss, Benedita Malheiro e Juan-Carlos Burguillo. Personalised placement in networked video. *Proceedings of the 21st international conference companion on World Wide Web - WWW '12 Companion*, page 959, 2012. URL: http://dl.acm.org/citation.cfm?doid=2187980.2188229, doi:10.1145/2187980.2188229.

[geo] geoawesomeness. Location Based Service - a little bit of theory. URL: http://geoawesomeness.com/location-based-services-a-little-bit-of-theory/.

[IAB16] IAB. Digital Video In-Stream Ad Format Guidelines and Best Practices [Report]. (May):4–23, 2016. URL: http://www.iab.com/wp-content/uploads/2016/01/DVAFG_2015-01-08.pdf.

[KD] J A N Kramoliš e Martina Drábková. The aspects of Product Placement as a marketing tool in the Czech Republic. pages 144–149.

[KD12] Jan Kramolis e Martina Drabkova. Types , Forms and Major Product Categories of Product Placement in the Czech Republic. 2012:11, 2012. doi:10.5171/2012.441984.

[KK11] Amit Kushwaha e Vineet Kushwaha. Location Based Services using Android mobile Operating System. *International Journal of Advances in Engineering & Technology*, 1(1):14–20, 2011. URL: http://www.ijaet.org/media/Location-Based-Services-using-Android-mobile-Operating-System-Copyright-IJ pdf.

[LCS+12] R E V W Illiam R C Laytor, Betsy Currie, Rhonda Stubbs, Nancy Smith, Jenna Robinson, A M Jerry Broome, Mark Mitchell, Steve Thrailkill e Ken Troutman. Video Ad Serving Template (VAST). page 71, 2012. URL: https://www.iab.com/wp-content/uploads/2015/06/VASTv3_0.pdf.

[Mir] Mirriad: Advertising for the Skip Generation. URL: http://www.mirriad.com/.

[MMCB16] Matthew Malloy, Mark Mcnamara, Aaron Cahn e Paul Barford. Ad Blockers : Global Prevalence and Impact. *Proceedings of the 2016 ACM on Internet Measurement Conference (ACM IMC'16)*, pages 119–125, 2016. doi:10.1145/2987443.2987460.

[MN09] Irena Mlýnková e Martin Nečaský. Towards Inference of more Realistic XSDs. *Proceedings of the 2009 ACM symposium on Applied Computing - SAC '09*, page 639, 2009. URL: http://dl.acm.org/citation.cfm?id=1529415, doi:10.1145/1529282.1529415.

[Pul14] Joe Pulizzi. The Ultimate Guide to Native Advertising. 2014. URL: https://www.linkedin.com/pulse/20140107180859-5853751-the-ultimate-guide-to-native-advertising.

[RDWG08] Ana-Maria Roxin, Christophe Dumez, Maxime Wack e Jafaar Gaber. Middleware models for location-based services. *Proceedings of the 2nd international workshop on Agent-oriented software engineering challenges for ubiquitous and pervasive computing - AUPC '08*, pages 35–40, 2008. URL: http://portal.acm.org/citation.cfm?doid=1387249.1387255, doi:10.1145/1387249.1387255.

[SLK12] M Zubair Shafiq, Alex X Liu e Amir R Khakpour. Revisiting Caching in Content Delivery Networks. *Sigmetrics*, pages 567–568, 2012. URL: http://homepage.divms.uiowa.edu/~mshafiq/files/cdn_sigmetrics.pdf, doi:10.1145/2637364.2592021.

[SM12] Maad Shatnawi e Nader Mohamed. Statistical techniques for online personalized advertising. *Proceedings of the 27th Annual ACM Symposium on Applied Computing*, pages 680–687, 2012. URL: http://dl.acm.org/citation.cfm?id=2245406, doi:10.1145/2245276.2245406.

[Sto11] Thomas Stockhammer. Dynamic Adaptive Streaming over HTTP: Standards and Design Principles. *Proceedings of the Second Annual ACM Conference on Multimedia Systems*, 2014(i):133–144, 2011. URL: http://doi.acm.org/10.1145/1943552.1943572, doi:10.1145/1943552.1943572.

[Swa13] Viswanathan Swaminathan. Are we in the middle of a video streaming revolution? *ACM Transactions on Multimedia Computing, Communications, and Applications*, 9(1s):1–6, 2013. URL: http://dl.acm.org/citation.cfm?doid=2523001.2490826, doi:10.1145/2490826.

[SWY08] Jungwon Min Shu Wang e Byung K Yi. Location Based Services for Mobiles: Techonogies and Stardards. (January 2008), 2008. URL: https://www.researchgate.net/publication/265758564_Location_Based_Services_for_Mobiles_Technologies_and_Standards.

[TG12] Christian Timmerer e C Griwodz. Dynamic adaptive streaming over HTTP: from content creation to consumption. *Proceedings of the 20th ACM international ...*, pages 1533–1534, 2012. URL: http://dl.acm.org/citation.cfm?id=2396553, doi:10.1145/2393347.2396553.

[VMBF16] Bruno Veloso, Benedita Malheiro, Juan C Burguillo e Jeremy Foss. Product Placement Platform for Personalised Advertising Product Placement Platform for Personalised Advertising. (December), 2016. URL: http://recipp.ipp.pt/bitstream/10400.22/8966/1/COM_LSA_NEMSummit_2016.pdf.

[WZX+16] Weihang Wang, Yunhui Zheng, Xinyu Xing, Yonghwi Kwon, Xiangyu Zhang e Patrick Eugster. WebRanz: web page randomization for better advertisement delivery and web-bot prevention. *Proceedings of the 2016 24th ACM SIGSOFT International Symposium on Foundations of Software Engineering - FSE 2016*, pages 205–216, 2016. URL: http://dl.acm.org/citation.cfm?doid=2950290.2950352, doi:10.1145/2950290.2950352.

[YF14] RGB Networks Yuval Fisher. an Overview of Http Adaptive Streaming Protocols for Tv Everywhre Delivery. 2014.

[ZAD+00] Jin Zeng, Oscar C Au, Wei Dai, Yue Kong, Luheng Jia e Wenjing Zhu. A Tutorial on Image / Video Coding Standards. page 7, 2000. URL: http://ieeexplore.ieee.org/document/6694346/.